

# Discrete-Time Demodulator Architectures for Free-Space Broadband Optical Pulse-Position Modulation

A. A. Gray<sup>1</sup> and C. Lee<sup>1</sup>

*The objective of this work is to develop discrete-time demodulator architectures for broadband optical pulse-position modulation (PPM) that are capable of processing Nyquist or near-Nyquist data rates. These architectures are motivated by the numerous advantages of realizing communications demodulators in digital very large scale integrated (VLSI) circuits. The architectures are developed within a framework that encompasses a large body of work in optical communications, synchronization, and multirate discrete-time signal processing and are constrained by the limitations of the state of the art in digital hardware. This work attempts to create a bridge between theoretical communication algorithms and analysis for deep-space optical PPM and modern digital VLSI. The primary focus of this work is on the synthesis of discrete-time processing architectures for accomplishing the most fundamental functions required in PPM demodulators, post-detection filtering, synchronization, and decision processing. The architectures derived are capable of closely approximating the theoretical performance of the continuous-time algorithms from which they are derived. The work concludes with an outline of the development path that leads to hardware.*

## I. Introduction

This work builds on a large body of previous work in optical communications, in particular, *Optical Communications*, by Gagliardi and Karp [1], “Design and Analysis of a First-Generation Optical Pulse-Position Modulation Receiver,” by Vilmrotter et al. [2], and “Design and Development of Deep Space Baseline Optical Transceiver,” by Yan and Chan [3], as well as many others that are referenced later in this work. The focus in this work is not on the analysis of performance of optical communications systems but rather on synthesis of discrete-time architectures suitable for realizing the functions of certain continuous-time processing using modern digital very large scale integrated (VLSI) circuits. The primary discrete-time signal processing building blocks are presented in this work along with preliminary performance results; the tasks required for synthesizing more complete demodulator architectures and analyzing their performances are outlined in the conclusion.

---

<sup>1</sup> Communications Systems and Research Section.

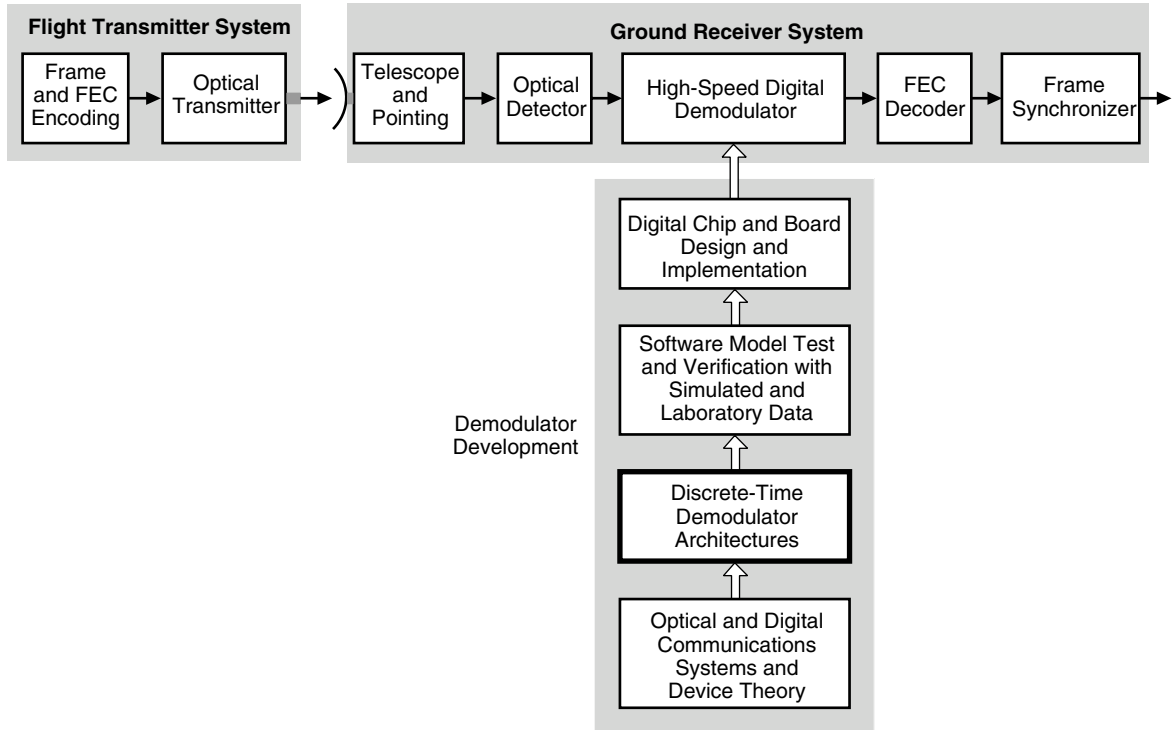
The research described in this publication was carried out by the Jet Propulsion Laboratory, California Institute of Technology, under a contract with the National Aeronautics and Space Administration.

Figure 1 presents a block diagram of the optical transmitter and ground receiver system. The development path of the digital demodulator also is indicated from a very simplified high level; a more complete description of this process is presented in the conclusion. The work presented here is one step in the development process, starting from theoretical formulation and ending with implementation of the digital demodulator. Many elements of the discrete-time architectures developed here are fundamental and remain valid independent of whether or not the communications channel is radio frequency or optical, and some of these elements are derived with the intention of overcoming limitations of state-of-the-art digital hardware.

The architectures are parameterized to utilize the tremendous flexibility achievable with modern digital hardware and to satisfy a large range of system requirements. System requirements play an increasingly critical role in the development as the steps in the process of Fig. 1 get closer to implemented hardware. Many system requirements may be met with numerous variations of the discrete-time architectures developed herein. The final determination of which architecture is fully developed to implementation and the derivation of numerous parameters are made from specific requirements, and they are constrained by implementation considerations such as commercially available high-speed chip and board technology. Next a brief overview highlighting the objective of each section is given.

Section II is an introduction to discrete-time architectures. The motivation for their development is discussed and includes the many advantages of implementation of signal processing using digital VLSI circuits.

Section III introduces discrete-time synchronization and slot filtering and introduces the impact of signal dynamics with fixed sample rate systems employing this processing. The motivation for fixed-phase analog-to-digital (A/D) sampling and the time-varying slot or interpolation filter is presented.



**Fig. 1. Block diagram of the system and simplified development path for the digital demodulator.**

Section IV extends the results of Section III to more complete demodulator models. Simplified models of demodulators are presented here, and these are extended throughout the remaining sections.

In Section V, problems identified in Sections III and IV are addressed by various signal processing methods, including non-linear processing. The methods for combining post-detection filtering and correction of synchronization errors are addressed with time-varying post-detection or interpolation filters.

Section VI presents parallel discrete-time demodulator architectures. The serial-processing results are extended to parallel or vector processing, which is required to achieve the processing necessary for broadband pulses requiring very high sample rates. This parallelization is performed on the core processing of slot-synchronization and post-detection filtering. Symbol synchronization and other demodulator algorithms are not explicitly addressed here but are considered much more straightforward to design in discrete time.

In Section VII, asynchronous discrete-time processing is addressed. The fixed-rate processing of a signal that contains modulated data with a rate that is asynchronous to the sample and system clocks creates significant challenges, specifically asynchronous digital design and implementation. The asynchronous processing is shown to create particular challenges for the parallel architectures.

Section VIII introduces representative discrete-time architectures derived using the methods, framework, and signal processing designs developed herein. In this section, we use generic parallel discrete-time algorithms and methods developed earlier and existing in the literature to synthesize specific designs that incorporate the capability for trading processing rate with complexity. We give the performance of a software model of a receiver architecture that includes a simplified model of the optical channel. While the demodulator architectures presented are not complete, they encompass many required functions, and the evolution to more complete architectures is presented in brief.

Section IX presents system models along with certain design/implementation equations. Frequency and digital data input/output (I/O) models are presented. The parallel discrete-time demodulator is described in terms of bandwidth and clock rates. This model is useful in establishing specific design parameters of the parallel digital demodulator and implementation platform from requirements such as data rates, pulse-position modulation (PPM) pulse bandwidth, and other requirements. The digital I/O model is useful for high-speed digital platform design by establishing the primary I/O requirements on the implementation platform.

The conclusion in Section X provides an overview of the work presented and indicates the next steps in the development of a broadband hardware implementation. These steps encompass discrete-time design and analysis, extensive software modeling and simulation, and hardware design tools and methods and realization. The conclusion includes an overview of the state-of-the art processes that lead to implementation in modern VLSI platforms and devices. Parts of this process are derived from targeting VLSI implementation in field programmable gate arrays (FPGAs). These devices have, in many instances, enabled improved development strategies of complex VLSI systems, reducing development risk while facilitating more aggressive schedules and development processes. The proposed development process incorporates extensive use of a variety of software models and tools, some of which are generic and used for system and subsystem validation and others that are device specific. Modern communications systems, such as the one developed here, generally are beyond any type of comprehensive closed-form analytical analysis or performance evaluation and rely extensively on Monte-Carlo simulation. However, as part of the process, analytical models and bounds are used extensively to validate various subsystem performances.

## II. Introduction to Discrete-Time Demodulator Architectures

Receivers are arguably the most complicated processing element (hardware or software) in a communications system. Modern digital receivers must be flexible enough to process parameterized modulation schemes, pulse shapes, and data rates. Receiver complexity is directly related, although not necessarily linearly, to the complexity of the modulation type used in the system. Many modulations developed for optical communication are complex, with some so complicated that it may be impractical to implement them in hardware. The PPM modulation is selected because it appears to be the modulation of choice for most future deep-space optical satellite communications systems. In addition, the demodulator architectures developed are based on the assumption of single-symbol decision processing, as opposed to symbol decisions that are based on maximum-likelihood sequence detection [4,5]. However, the demodulator may process input signals to generate soft symbols at the output of the post-detection filter that may be further processed by a forward error-correction decoder (maximum-likelihood sequence decision processing).

With the availability of digital VLSI technology, coupled with the flexibility of discrete-time signal processing algorithms, it is very desirable to implement receivers with as much digital processing capability as possible [6,7]. Dynamic range in filter bandwidths of greater than 8 orders of magnitude is feasible, and similar flexibility exists with other processing functions when implemented in modern digital circuits. In addition, an all-digital demodulator implemented using complementary metal-oxide semiconductor (CMOS) technology has great advantages in size, reliability, and greatly reduced reproduction costs over analog demodulators as well as other digital technologies [8,9]. The number of options for realizing digital VLSI circuits is extensive and includes many field programmable gate arrays (FPGAs), commercially available digital signal processors (DSPs), and application-specific integrated circuits (ASICs). FPGAs in particular have increased the flexibility of high-speed digital processing, with large-scale reconfiguration of VLSI possible with relatively rapid design cycles.

Often the only true limitation to the data rates an all-digital demodulator can process is the analog-to-digital converter. However, conventional CMOS digital demodulators currently have substantially lower clock rates than the fastest commercially available analog-to-digital converters. In such a high-rate system, the minimum number of samples required for discrete-time processing, the Nyquist rate, is required to process the maximum data rate possible, the Nyquist data rate. Because commercially available CMOS hardware generally has lower clock rates than the fastest A/D converters, CMOS digital demodulators using traditional serial algorithms for digital communications process data rates many times lower than the maximum practical Nyquist data rate. The rate difference motivates the development of parallel-processing architectures.

It should be noted that, strictly speaking, there is no such thing as an all-digital receiver. There is always some analog processing for the optical-to-digital conversion typically accomplished by optical-to-electrical (analog) conversion and then conversion to a digital signal. For the purposes of this work, the definition of an all-digital demodulator is one in which the modulated optical waveform is detected and converted to an analog current and then to a voltage signal. This voltage is sampled, and the demodulator functions of post-detection filtering, symbol decision processing, synchronization (including slot timing and symbol recovery for PPM), and symbol-to-bit conversion are performed using exclusively digital processing; the continuous-time A/D clock phase and frequency are not adjusted using feedback from the digital demodulator. There are many other functions of a demodulator that, although necessary, will prove more trivial to implement and are not dealt with in this work. These functions include slot and symbol synchronizer lock detection, power estimation, error-control decoder preprocessing, and others. Here we derive discrete-time architectures for realizing the primary demodulator functions using known continuous-time processing, acknowledging that the processing utilized in a demodulator and the performance obtained are highly dependent on the optical channel used in the system. We assume band-limited photon-counting detection and the channel conditions outlined in [2]. In the architecture development,

we place particular emphasis on the two greatest and related challenges of a Nyquist data rate digital implementation, Nyquist rate sampling of communications signals and very high rate (bandwidth) processing.

### III. Discrete-Time Synchronization and Post-Detection Filtering Overview

The PPM slot and symbol synchronization, post-detection filtering of the input signal, and symbol decision processing form the core of the optical PPM satellite communications receiver. Synchronization and post-detection filtering often are thought of as separate operations; however, we will demonstrate here that they may be combined in an intuitive way to create time-varying discrete-time filter architectures. Furthermore, these architectures are well suited for and motivated by all-digital implementation.

The post-detection filter is a specified operation that is used to process the received signal in a precise time-synchronized fashion. The incoming waveform or received signal often is convolved with the post-detection filter, although there are other types of processing employed depending on the system [1]. In the development of the discrete-time architecture here, we assume that the incoming sampled waveform is convolved with a post-detection filter. The waveform analysis is greatly simplified and ignores the statistical nature of the signal at the output of the optical detector. This simplified analysis should be thought of as a conceptual tool to reveal many fundamental elements of the parallel-processing discrete-time architecture design that are largely independent of the statistical nature of the input signal or even of whether the channel is radio frequency or optical. This approach results in a general architecture for accomplishing time-synchronized processing that encompasses or is readily extended to other processing that does incorporate the detailed nature of the input signal. Regardless of the post-detection filter coefficients or even if it is not a linear filter/convolution, precise time synchronization of the required signal processing in the presence of signal dynamics is generally required. Understanding the implications and properties of this precise time synchronization with Nyquist or near-Nyquist sampling motivates much of the discussion and analysis here; this understanding is critical in developing discrete-time demodulator architectures for Nyquist data rates. The results of this section are used in Section VIII to develop architectures with generalized slot-synchronization and post-detection filtering.

#### A. Discrete-Time Post-Detection Filtering

In the generic digital communications transmitter, baseband bits are mapped to modulation symbols,  $A_k$ . The transmit pulse shape is given by  $p(t)$ . The output of the transmitter is then

$$Q(t) = \sum_{k=-\infty}^{\infty} r_k p(t - kT_{sym} - A_k T_{slot} + \phi_k) \quad (1)$$

Here  $T_{sym}$  is the symbol period,  $T_{slot}$  is the slot period,  $r_k$  represents the intensity variations, and  $\phi_k$  is a timing jitter sequence [2]. The timing jitter and intensity variation terms will be excluded from the simplified analysis and discussion here. As an example, consider the 8-PPM signal set in Fig. 2. Transmitting the signal,  $Q(t)$ , through a linear noiseless channel with impulse response,  $b(t)$ , will result in a received signal

$$R(t) = \int_{-\infty}^{\infty} b(\tau) \sum_{k=-\infty}^{\infty} p(t - kT_{sym} - A_k T_{slot} - \tau) d\tau \quad (2)$$

This can be rewritten as

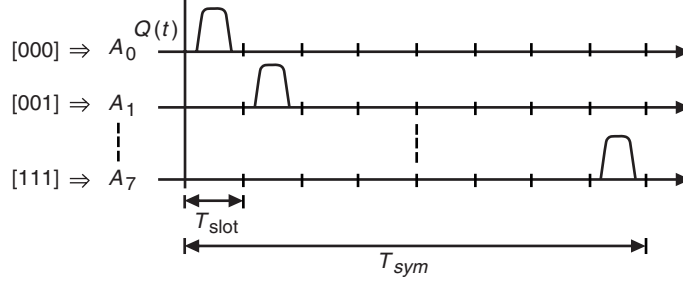


Fig. 2. Example PPM symbol set.

$$R(t) = \sum_{k=-\infty}^{\infty} h(t - kT_{sym} + A_k T_{slot}), \quad h(t) = \int_{-\infty}^{\infty} b(\tau) p(t - \tau) d\tau \quad (3)$$

where  $h(t)$  is the received pulse. The result of the continuous-time input filtered with some post-detection filter is then

$$y(t) = \int_{-\infty}^{\infty} R(t - \tau) f(\tau) d\tau \quad (4)$$

where the filter  $f(t)$  may be designed from some optimality criteria or criterion derived for the optical channel. Here we derive the time-varying slot filter assuming a classical matched-filter approach common in radio-frequency communications; this derivation will illustrate fundamental problems with discrete-time synchronization for which architectures then are designed to solve. The post-detection filter peak signal-to-noise ratio (SNR) or optimal output is obtained by sampling at the slot rate,

$$y_M(n) = y(T_{slot}n), \quad n = 1, 2, 3, \dots \quad (5)$$

It should be noted that, in an operational wireless communications system with channel noise, including timing jitter, the desired signal or sample generally is not obtained, but some estimate or approximation is. For example, with jitter  $\phi_k$  that varies from one pulse to the next, the optimal sample estimate  $\hat{y}_M(n)$  generally will not equal the true optimal sample point,  $\hat{y}_M(n) \neq y_M(n)$ . If the system is designed properly, the estimate of the optimum sample (or samples) will have zero average error, assuming there are no other unaccounted for dynamics or signal distortions and it is an unbiased estimator [10–12]. We progress assuming no jitter in the input signal so as to derive necessary fundamental concepts in a simplified fashion, but it must be understood that the slot-synchronization algorithm design should average the effects of jitter and other channel noise sources in such a way that an unbiased estimate of timing offset is obtained and then corrected for. For a discrete-time demodulator system with sample period  $T_s$ ,

$$h_d(n) = h(T_s n + \Delta t_R) \quad (6)$$

$$f_d(n) = h(-T_s n - \Delta t_M) \otimes K(T_s n) \quad (7)$$

$$R_d(n) = \sum_{k=-\infty}^{\infty} h_d(n - kT_{sym} + A_k T_{slot}) \quad (8)$$

$$y_d(n) = \sum_{k=-\infty}^{\infty} R_d(k) f_d(n-k) \quad (9)$$

Here  $\Delta t_R$  is the time offset of the received signal  $h_d(n)$ , and  $\Delta t_M$  is the time offset of the post-detection filter  $f_d(n)$ . In general, for  $f_d(n)$  to have equivalent performance to the continuous-time post-detection filter, the following condition must be met:  $\Delta t_M = \Delta t_R$  (other conditions, such as band-limited signals, also must be met). If this condition is not met, the discrete-time system realizing  $f_d(n)$  generally will result in different (probably reduced) performance as compared with the continuous-time system with  $f(t) = h(-t) \otimes K(t)$  [5]. Note that  $K(t)$  is a designed impulse response derived from system-specific criteria; the simplified case is an impulse and could perform the same role as a constant for scaling. So  $f(t)$  is a convolution of the time-reversed received pulse and a design pulse  $K(t)$ . The output of the discrete-time filter may be downsampled at the slot rate

$$y_D(n) = y_d(Dn) \quad (10)$$

Obtaining synchronized samples at the output of the post-detection filter is required to obtain the optimum filter output defined by some criterion. Obtaining this correct sample or samples (if no downsampling or another downsample rate is used) is the objective of slot synchronization. Any sampling offset [13] may result in performance degradation; this performance degradation may occur in varying degrees to the decision process and the slot-synchronization algorithm itself as some closed-loop slot-synchronization algorithms contain the post-detection filter in their feedback path [9]. Figure 3 illustrates a conceptual (post-detected) received waveform and post-detection filter demonstrating ideal and non-ideal slot timing at some time  $t_0$ . Note that if  $\Delta t_M \neq \Delta t_R$  the peak will not be represented by a discrete-time sample, but the error may be removed or minimized via a variety of discrete-time processing methods; however, these may entail significant challenges to realize in real-time hardware with very high rate sampled systems.

In this conceptual example, an arbitrary slot or receive filter was used. This could just as well be an integrate-and-dump filter (a moving average filter with downsample), in which the signal may be synchronized before the integrate-and-dump by some time-varying interpolation filter to adjust the phase of the signal (an example of this is given in a later section). Recall that, in a discrete-time communications receiver with very high data (sample) rates, when flexibility is required or when the receiver is used for ranging, it is very desirable to fix the sample clock; in addition, adjusting the oscillator may be difficult or impractical in near-maximum Nyquist data rate receivers due to the very high clock rate, much greater than 1 GHz with modern A/Ds. However, when sampling the received signal at a constant sampling

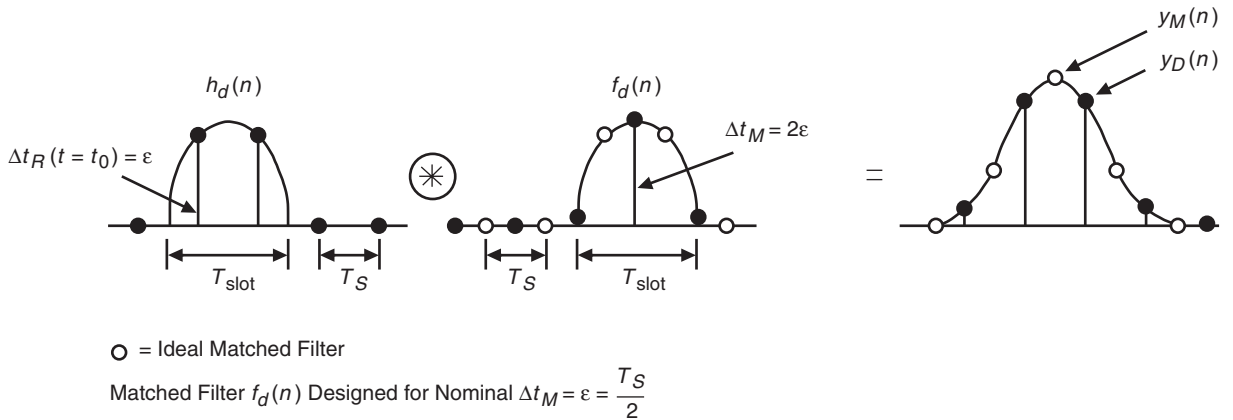


Fig. 3. Illustration of discrete-time post-detection filtering.

rate  $f_s$ , the sample offset  $\Delta t_R$  defined earlier changes with time—that is,  $\Delta t_R(t)$ . This is due to the non-linear Doppler channel created by the relative velocity between ground stations and the transmitting satellite and the difference between the transmitter and receiver clocks. These differences are impossible to predict precisely. The difference between the discrete-time post-detection filter output and the ideal continuous-time post-detection filter output is the fractional sampling offset. The fractional sampling offset (or simply the sampling offset) can be no larger than one-half of a sample period, or  $T_s/2$ . For the downsampled discrete-time post-detection filter output to be equal to the continuous-time matched-filter sampled output,  $y_D(n) = y_M(n)$ , from the previous equations,

$$\begin{aligned} y_D(n) &= y_d(nD) = \sum_{k=-\infty}^{\infty} \left( \sum_{m=-\infty}^{\infty} h(T_s k - mT_{sym} + A_m T_{slot} + \Delta t_R) f(nT_s D - k) \right) \\ &= \int_{-\infty}^{\infty} \left( \sum_{m=-\infty}^{\infty} h((nT_{slot} - \tau) - mT_{sym} + A_m T_{slot}) \right) f(\tau) d\tau = y_M(n) \end{aligned} \quad (11)$$

This equality cannot be valid unless the following condition is met:

$$nT_s D + \Delta t_R = nT_{slot} \quad (12)$$

Note that  $\Delta t_R$  is varying with time, and so is  $T_{slot}$ , but in a continuous-time processing system it is assumed that the oscillator frequency and phase are adjusted such that the matched-filter output sampling time  $T_{slot}$  is optimal. Recall that  $T_s$  is fixed for the discrete-time system proposed. This fact, combined with a time-varying  $\Delta t_R$  with fixed  $\Delta t_M$ , implies that the discrete-time matched-filter peak, or other optimal sample or samples if another optimality criterion is used, in general will not be represented by the discrete-time samples of the post-detection filter output. Furthermore, it should be noted that, when Nyquist rate sampling is utilized, the sample period is the largest possible, making the potential sampling offset the largest possible. The A/D oscillator phase theoretically could be adjusted such that  $\Delta t_R$  is obtained to match  $\Delta t_M$ .

The slot-synchronization algorithm estimates the slot time delay ( $\Delta t_\epsilon = \Delta t_M - \Delta t_R$ ) and frequency  $1/T'_{slot}$  from the received signal  $R(t)$ , where the true slot frequency is  $1/T_{slot}$ . Slot synchronization for the optical channel may be accomplished by a number of algorithms ranging from optimal estimators to highly sub-optimal estimators and may be an open-loop estimator or closed-loop estimator with feedback. It turns out in a discrete-time or digital implementation with Nyquist-rate sampling at very high bandwidths that detecting or estimating the error often is less challenging than correcting for it in real time. While the type of estimator and its estimation performance are key considerations in any demodulator design and implementation, a nominal algorithm is chosen from [3], and the development presented here is focused primarily on how to correct for time-varying synchronization errors that the algorithm estimates. It will become apparent that many known estimation algorithms can be used in conjunction with the architecture developed with proper modification and/or discrete-time design. However, determining the performance of the resulting discrete-time system may be very challenging.

A property of the PPM demodulator is that once slot synchronization is estimated and the slot clock determined, then the symbol clock generally may be readily derived, and hence the bit clock also may be derived. Symbol synchronization is required to determine symbol boundaries so that the slot position of the pulse in the symbol can be estimated; the estimate  $\hat{A}_k$  can be made. This process is referred to as the decision process; refer to Fig. 4. Following this estimate, symbol-to-bit mapping is performed; this operation inverts the bit-to-symbol mapping at the transmitter. Following this processing, forward error-correction decoding may be performed if coding is used in the system. In modern systems, the



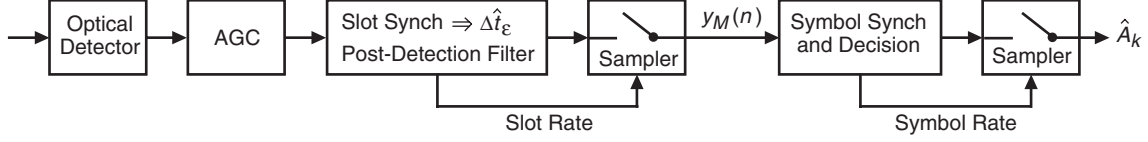


Fig. 4. Simplified primarily continuous-time receiver.

process of symbol-to-bit mapping and decoding may be performed in a coupled fashion. A simplified optical communications receiver is illustrated in Fig. 4. Note the two-step symbol process: first, slot synchronization (slot clock recovered) and pulse filtering are performed on all slots to determine  $y_M(n)$ ; second, symbol synchronization and symbol decision processing are accomplished to form the symbol estimates  $\hat{A}_k$ . Other processing follows, such as forward error correction and frame synchronization.

Figure 5 illustrates a simplified discrete-time receiver with discrete-time post-detection filter. This system employs a constant clock rate and synchronous processing and, therefore, is a greatly simplified model. The slot-synchronization algorithm recovers the true slot clock by estimating slot phase and frequency offset from the nominal slot clock; this nominal clock frequency is  $1/T_s$ . By obtaining the estimate  $\hat{\Delta t}_\varepsilon$  of  $\Delta t_\varepsilon$ , the phase of the slot clock is determined and the frequency also may be estimated using successive  $\hat{\Delta t}_\varepsilon$  estimates. The number of samples per slots for this signaling is  $D = T_{\text{slot}}/T_s$ , the slot period divided by the sample period. The number of samples per PPM symbol is  $C = DP$ , where  $P$  is the number of slots per symbol, that is,  $P = 2^m$ . Next we will analyze the critical deficiencies in this model and use the results to develop appropriate discrete-time signal processing algorithms and architectures to overcome these deficiencies.

If the Nyquist criterion is met, the peak, or any other sample(s), may be obtained or estimated using discrete-time phase-delay methods [14,15]. One method for realizing the latter is to change the discrete-time slot filter,  $f_d(n)$ , with time to match the time-varying incoming sampled received pulse shape such that  $\Delta t_\varepsilon(t) = \Delta t_R(t) - \Delta t_M(t) \approx 0$  or with noise sources in the system the mean estimate of  $\bar{\Delta t}_\varepsilon(t) = \bar{\Delta t}_R(t) - \bar{\Delta t}_M(t) < \varepsilon_0$  with variance  $\text{var} \{ \bar{\Delta t}_\varepsilon(t) \} < \varepsilon_1$ . Another method is to use a generic time-varying interpolation filter. Therefore, in a discrete-time system, slot synchronization can be thought of as having two components. The sampling offset must be removed ( $\Delta t_M(t) = \Delta t_R(t)$ ) or minimized to ensure that one (or more) of the discrete-time samples, determined by some selection criterion, represents the slot energy in the decision process. The discrete-time architecture must minimize the fractional sampling offset in order to approximate the performance of the ideal continuous-time system.

Static timing or phase offset has been considered under the assumption the received signal phase has first and second derivatives that are zero. This is just the first step in developing effective demodulator architectures, but it should be realized that in an operational system with scenarios including Doppler the first and second derivatives of phase generally are not zero (although higher derivatives often can be ignored [16]). There are frequency offsets and frequency rate changes. These correspond in not only a time-varying phase shift of the post-detection filter but also possibly in a change in the post-detection filter impulse response itself as the pulse, slot, and symbol periods either expand or contract with Doppler.

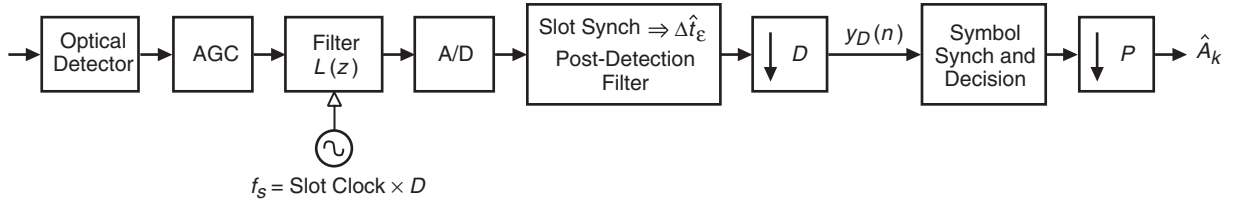


Fig. 5. Simplified discrete-time receiver.

These non-linear pulse distortion considerations are beyond the scope of this work. However, it will be become clear that once the basic framework for developing, analyzing, and implementing a discrete-time architecture with time-varying phase capability is completed, extending it to higher-order dynamics is possible and is very similar to previous work in the field of synchronization.

In summary, we have demonstrated that, in a discrete-time system with signal dynamics and a fixed sample clock, slot-synchronization/post-detection filtering poses challenges to discrete-time design and implementation, particularly with Nyquist sampling. These concepts are considered further in Section V in the development of discrete-time architectures for accomplishing synchronization error correction for broadband pulses.

## B. Slot and Symbol Synchronization and Decision Processing

Here, a brief summary is presented of known signal processing for performing slot and symbol synchronization, post-detection filtering, and decision processing outlined in [1–3] for a simple optical channel model. References for previous work include [17–33]. Some of the structures presented here do not represent unique or optimal solutions but may represent good design choices based on analysis and systems presented in the references given. Figure 6 illustrates the conceptual continuous-time slot synchronization, a variation of which is analyzed in [34].

The discrete-time or sampled version of Fig. 6 is given in Figs. 7 through 9; to date, performance of the discrete-time system has not been analyzed in closed form. The filter structure is the generic discrete-time version (using the impulse invariant transformation) of the continuous-time closed-loop second-order loop filter used commonly in phase-locked loops, Costas loops, data transition tracking loops, early-late gate-type loops, and many other synchronization loops. The performance and loop-filter design must be determined through analysis analogous to that performed numerous times on similar synchronization loops that exist both in published research and analysis and in operational systems, assuming the additive white Gaussian noise channel; examples of these derivations may be found in [35,36]. This filter structure is derived from the two-pole continuous-time loop and may be used to track both phase and frequency. Note that the loop filter  $L(z, l)$  in Figs. 8 and 9 may be extended to include more poles (increasing the order of the loop) and is a function of  $l$ . The latter characteristic indicates the ability to switch on-the-fly the filter bandwidth, which is commonly done in communications systems. The filter coefficients may be changed on-the-fly to achieve different bandwidths or damping factors and to increase performance during the various regions of operation of the receiver. For example, tracking bandwidths are often different from acquisition bandwidths, and it may be desirable to change bandwidths or even tracking-loop order with signal-to-noise ratio and/or signal dynamics. Obviously, an intelligent controller, such as a state-machine or human-in-the loop, must be used to control such changes.

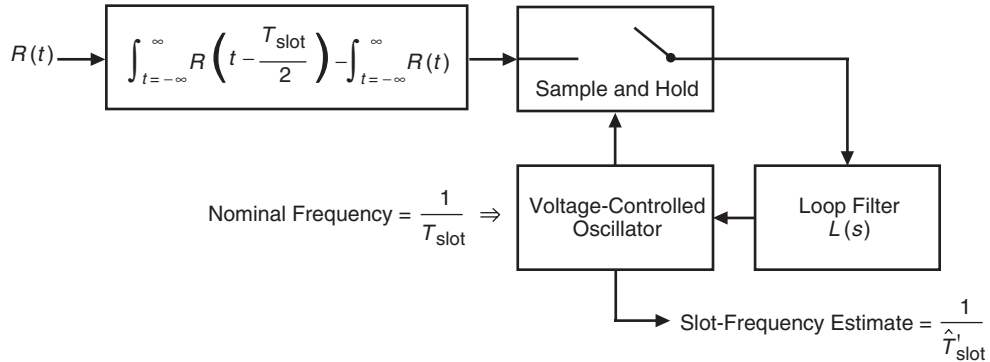


Fig. 6. Continuous-time model of the slot-synchronization loop.

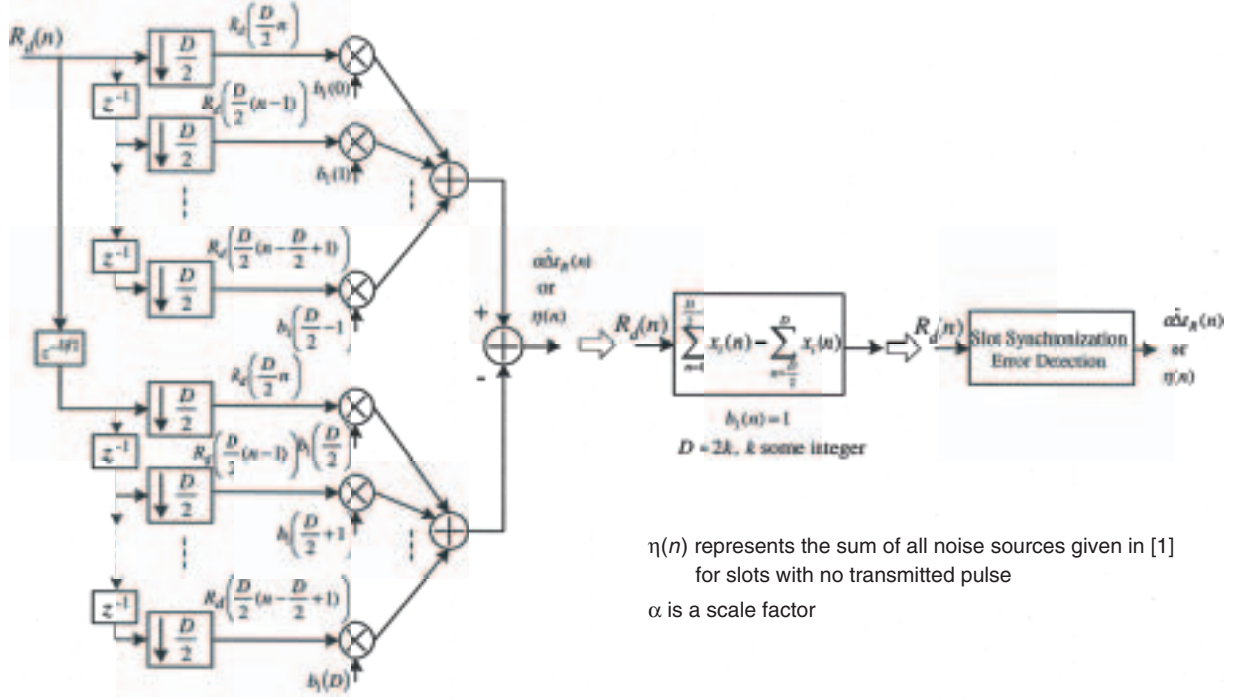


Fig. 7. Generalized and simplified discrete-time slot-error detection.

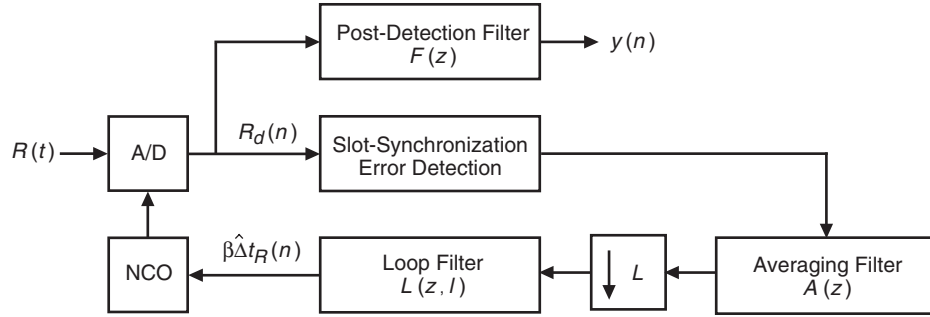


Fig. 8. Discrete-time slot-synchronization loop.

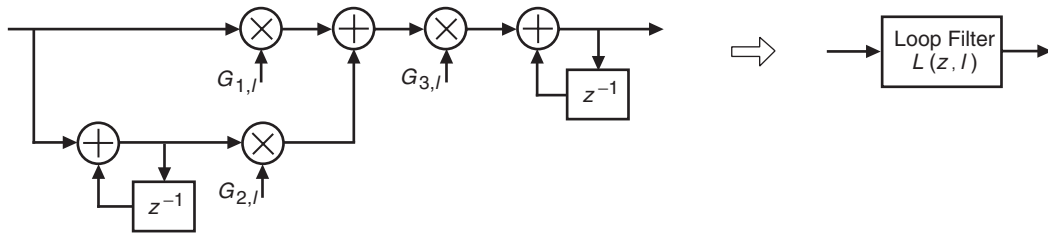


Fig. 9. Discrete-time and time-varying loop filter structure.

Figure 10 illustrates the operations of symbol synchronization and decision processing. The various algorithms fall into two general categories, those requiring a pilot sequence to acquire and track and those that operate on random data sequences, the so-called blind acquisition and tracking algorithms. There is relatively little previous work on evaluating the performance of these algorithms with the specific optical channel as described in [1,2]. It is assumed in this development that an algorithm using a periodic pilot sequence will be utilized, making the acquisition and tracking of a large range of PPM orders relatively straightforward.

From [1], the post-detection and decision processing are highly dependent on the characteristics of the optical channel and the optical detector used. Given the channel and operating scenarios outlined in [2], the slot-synchronized decision process of choosing the slot with the greatest energy from the slots within a symbol boundary to estimate which symbol was transmitted (the so-called energy detect receiver, whereby energy is used to estimate photon count) is nearly optimum for a range of system parameters. This is fortunate as this decision processing is readily accomplished with digital processing and may be easily parallelized. Figure 11 illustrates a simple example of the detection and decision process. Here ideal slot and symbol synchronization are assumed. The filter  $F(z)$  is a simple moving average filter with  $D$  coefficients ( $f(n) = 1$ ), where  $D = 4$  is the number of samples per slot, and the output is downsampled by  $D$ . The output of the downsampler is the sum of all the samples in a slot; these then are processed to decide which slot has the greatest energy. The samples in the slots other than the slot containing the transmitted pulse are non-zero due to the noise on the input signal.

Although symbol synchronization and detection/decision processing are essential processing elements and their performances must be evaluated carefully in any theoretical or practical receiver development, they generally are not as challenging to realize in digital VLSI as are slot synchronization and post-detection filtering. These former functions generally are processed at much lower rates, depending on PPM order, than slot filtering and timing error estimation and correction processing. Of course, if other types of decision algorithms that require more complex processing are needed, this may not be the case.

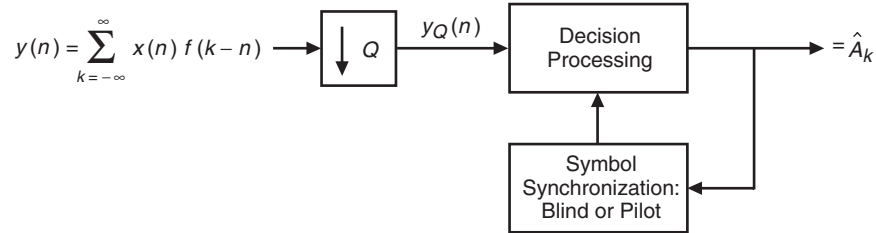


Fig. 10. Discrete-time symbol synchronization and decision processing.

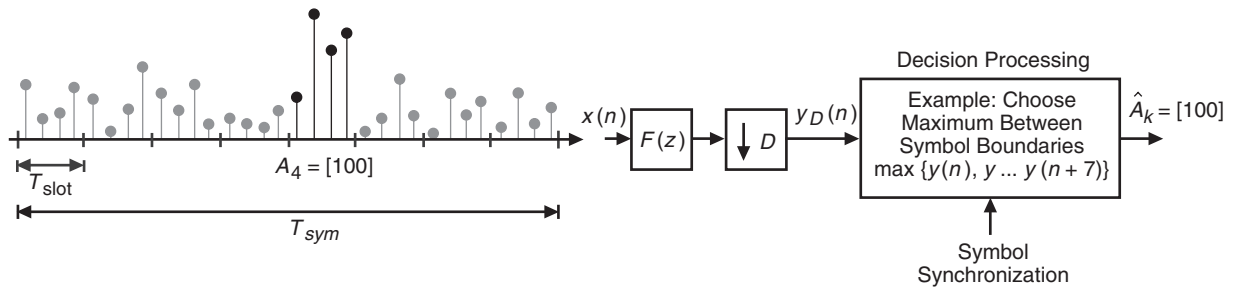


Fig. 11. Example symbol synchronization and detection,  $Q = D$  and choose maximum detection.

Finally, as stated previously, many of these algorithms are not uniquely optimum but are dependent on the channel characteristics, dynamics, etc. The algorithms presented very briefly in this section are meant to be representative of the signal processing structures used for accomplishing slot timing synchronization, post-detection filtering, symbol synchronization, and decision processing. One of many possible modifications is the integrated symbol- and slot-synchronization algorithm that may offer increased performance, particularly with high-order PPM. The structures presented may be readily modified to accommodate the feedback from symbol synchronization to slot synchronization, and an example of this type of structure is given in Section VIII; however, the performance of such a system for the optical channel has not been established to date. The optimal slot and symbol synchronization and detection are highly dependent on the optical detector and other channel characteristics; this provides motivation for highly flexible—that is, parameterized and reconfigurable—discrete-time processing architectures. We now introduce demodulator variations based on these “baseline” algorithms in a more global context and identify significant design choices along with specific challenges to their realization in discrete-time or digital VLSI.

#### IV. Discrete-Time Demodulator Variations

A discrete-time demodulator for PPM-type modulation is illustrated in Fig. 12. Based on the definition of the all-discrete-time demodulator developed in this work, Fig. 12 is not strictly an all-discrete-time demodulator but a hybrid demodulator, as the slot-timing recovery is performed with continuous-time processing. Figure 13 is a variation on this same theme but incorporates more discrete-time processing. Figure 14 illustrates an all-digital demodulator for a communications system with time-varying slot phase and period and may represent a reasonable design choice for some systems. For example, this demodulator may provide adequate performance in systems with many more samples per slot than required by the Nyquist rate. Nyquist rate sampling in such a system implies that some additional discrete-time signal processing is necessary to synthesize continuous-time phase adjustments of the A/D converter in order to approximate performance of the system in Fig. 13.

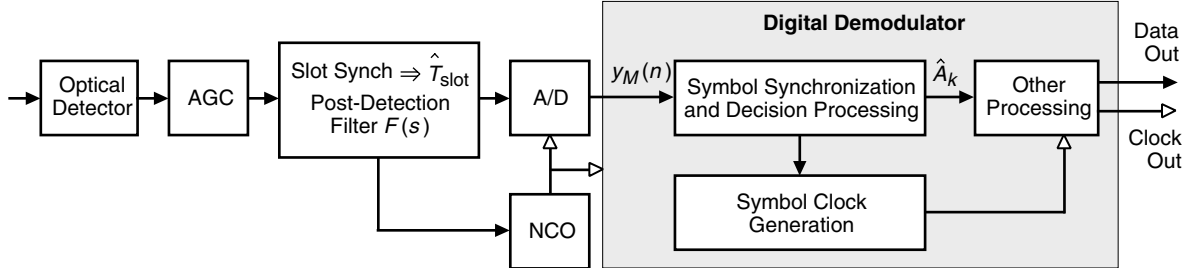


Fig. 12. Hybrid analog and discrete-time PPM receiver.

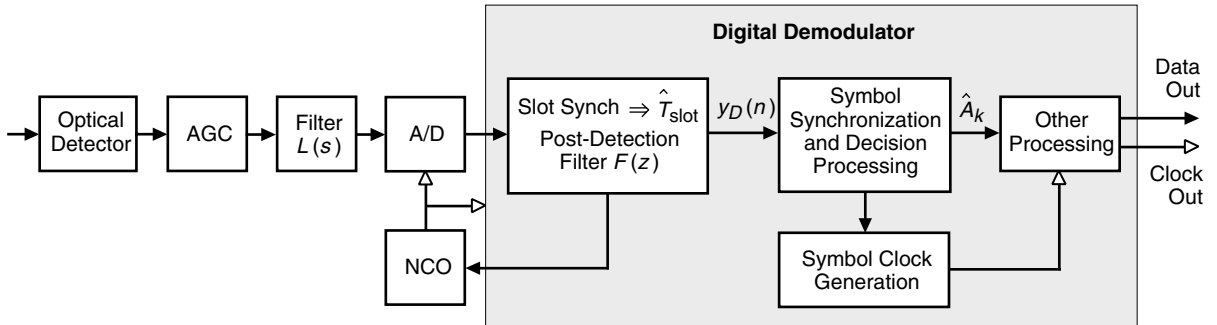


Fig. 13. Discrete-time demodulator with feedback to A/D.

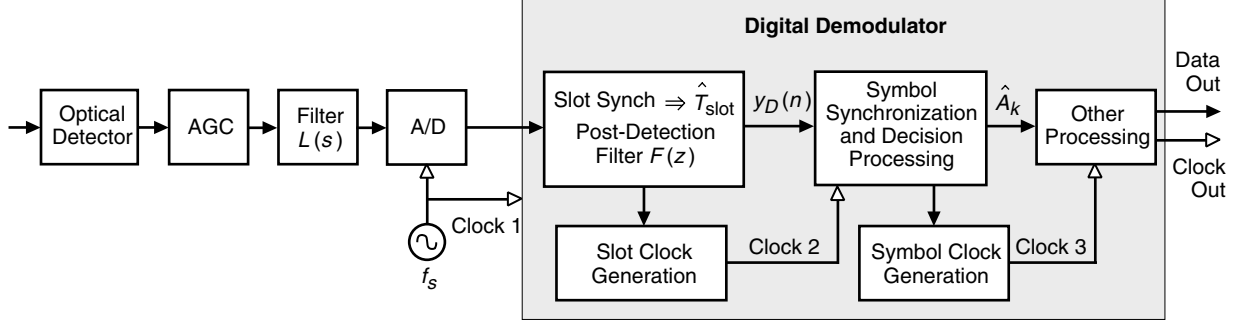


Fig. 14. All discrete-time demodulator.

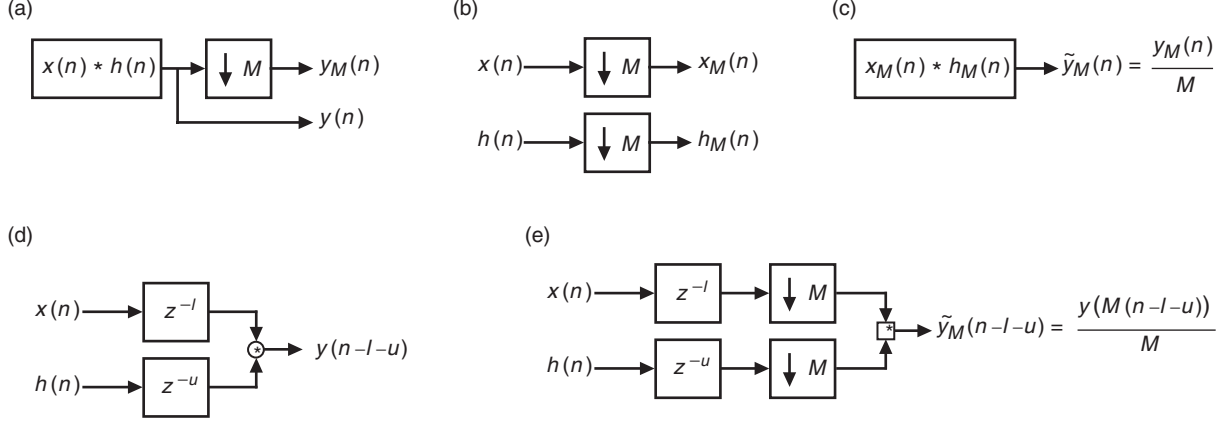
For very high sample rates, ranging applications where an ultra-stable clock source may be desired or a large dynamic range in data rate or input bandwidth is desired, or if other flexibility (high parameterization or reconfigurability) is desired, the receiver in Fig. 14 is the desired implementation [9]. At very high sample rates ( $>1$  GHz), adjusting the A/D oscillator phase to correct for synchronization errors as in Figs. 12 and 13 may be very difficult. The analog circuits necessary to accomplish this function are also generally very limited in dynamic range of bandwidths, signal amplitude variations, etc. The architecture in Fig. 14 is the baseline all-discrete-time demodulator architecture that we will further develop to mitigate losses in slot-synchronization performance, post-detection filtering, and decision processing due to a fixed-sample rate near Nyquist rate sampling. In addition, to operate at very high sampling rates ( $>1$  GHz), parallel processing is required given the current state of the art in commercial-off-the-shelf (COTS) CMOS VLSI circuits.

Finally, the three receivers in Figs. 12 through 14 are greatly simplified versions of actual receivers and represent the variations of primary interest from a top-level architecture development. There are many additional variations, functions (many diagnostic in nature) necessary in an operational receiver. However, Fig. 14 represents the core of the most challenging discrete-time processing that must be further developed to operate with Nyquist-rate sampling and with processing rates 10 to 20 times slower than the required A/D converter rates for sampling broadband PPM ( $>500$ -MHz bandwidth).

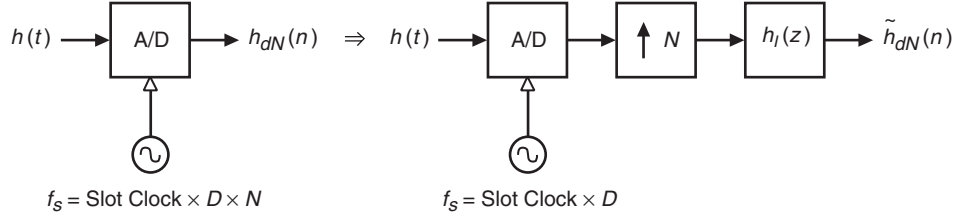
## V. Discrete-Time Demodulator with Time-Varying Post-Detection Filter

Now we derive a serial system such as that illustrated in Fig. 14 with a nominal sample rate  $D(1/T_{\text{slot}})$  such that, with Nyquist or near-Nyquist sampling, the system has the performance of a system employing a sample rate many times the Nyquist sample rate—that is to say, a system with sample rate  $D(1/T_{\text{slot}})$  that operates with identical or nearly identical performance to the design with sample rate  $DN(1/T_{\text{slot}})$ , where  $N$  is some large positive integer (for example  $>100$ ). Here the assumption is made that the received pulse is full response and band limited (which is impossible in theory but can be approximated in practice). For the scenario in which the signal is band limited but not full response [4], the system developed will not be identical but may be designed to be nearly equivalent. Several properties of multirate systems relevant to this development are illustrated in Figs. 15(a) through 15(e) [14].

Figure 15(a) illustrates some discrete-time sequence  $x(n)$  convolved with another discrete-time sequence  $h(n)$ . Figure 15(b) illustrates the notation for the downsampled sequences. If  $x(n)$  and  $h(n)$  are perfectly band limited such that no aliasing results from this downsampling (implying infinite sequences), the result of this convolution is related to  $y_M(n)$ , as demonstrated in Fig. 15(c), by a constant scale factor,  $M$ . If  $x(n)$  and  $h(n)$  are not infinite in time extent, convolving their downsampled versions as in Fig. 15(c) will result in a sequence that can be related to  $y_M(n)$  by some distortion function that incorporates the effects of aliasing caused by the downsampling. Under the assumption of this development, there is no aliasing. Figures 15(d) and 15(e) illustrate the effect that independent sample delays have on the convolved output and how such a delay relates to  $y(n)$ . As illustrated in Fig. 16, the sampled



**Fig. 15. Multirate signal processing properties: (a) downsampled convolution, (b) downsampled input, (c) convolved downsampled signals, (d) convolved time-shifted signals, and (e) convolved time-shifted and downsampled signals.**



**Fig. 16. Approximating high-sample-rate system with low-sample-rate system.**

received pulse shape  $h_{dN}(n)$  can be approximated from  $h_d(n)$  using the expander and interpolation filter. If we make the simplifying assumptions that  $h(t)$  is perfectly band limited (with  $D$  samples per slot) and full-response (although theoretically not possible) and the perfect interpolation filter is used (infinite time extent) [14], then  $\tilde{h}_{dN}(n) - h_{dN}(n) \approx 0$ .

Now the time-variable slot or post-detection filter bank ( $f_d(n, u)$ ) can be developed from the received pulse,  $h_{dN}(n)$  (or  $\tilde{h}_{dN}(n)$ ), as illustrated in Fig. 17 [15]. The filter bank  $f_d(n, u)$  may be developed from the time reversed  $h_{dN}(n)$  or another filter  $v(n)$ . The filter  $v(n)$ , for example, could be an interpolation filter. Each sampled filter  $u$  is shifted in time by  $-T_S/N$  from  $u - 1$  (recall the sample period of  $T_S$  and  $D$  samples per slot).

It is clear from this development and the properties in Figs. 15(c) through 15(e) that the output of Fig. 18(b) is equivalent to the output of Fig. 18(a) divided by the scale factor  $N$  [15]. Assuming the delays  $l$  and  $u$  are correct, such that the output is the desired sample (possibly with residual fractional sampling offset,  $\Delta t_e \leq T_S/2N$ ), any sample that can be obtained with the  $DN(1/T_{\text{slot}})$  sampled system can now be obtained from the  $D(1/T_{\text{slot}})$  sampled system due to the bank of time-shifted filters.

These properties are combined to generate the architecture in Fig. 19 that utilizes primarily synchronous processing and can operate with Nyquist rate sampling while approximating the performance of a system with a much higher sample rate ( $N$  times more samples per slot).

There are many other aspects of the time-varying post-detection filter and slot-synchronizer design that require analysis and further development on the path to practical implementation and to determining performance with realistic channel conditions. There is one issue in particular that requires further careful consideration in both the architecture development and hardware design and implementation. Recall that

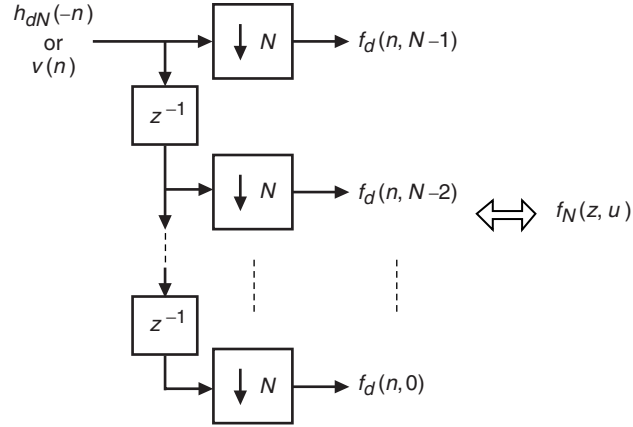


Fig. 17. Filter bank generation.

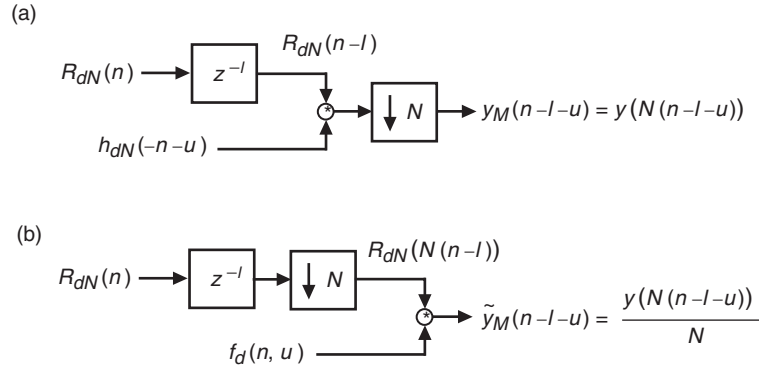


Fig. 18. Multirate signal processing properties: (a) delay, convolution, and downsample, and (b) delay, downsample, and convolution.

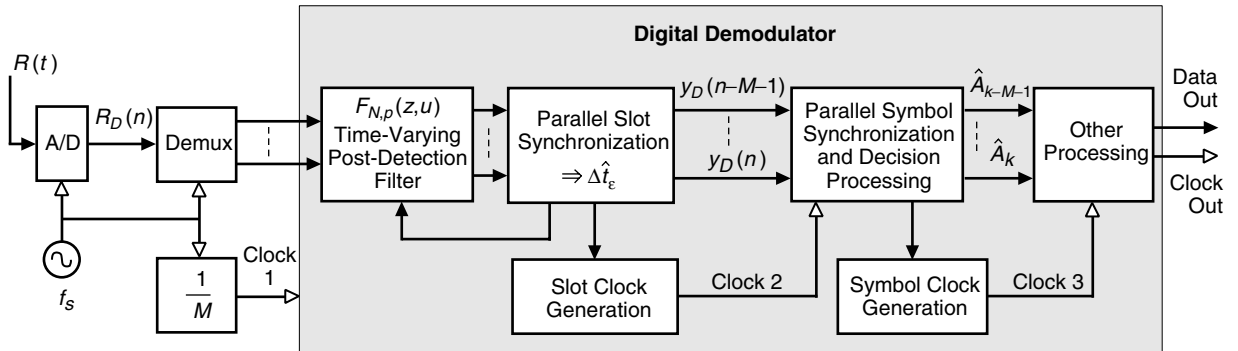


Fig. 19. Nyquist-rate discrete-time demodulator.



the reason that post-detection filter  $F(z)$  needs to vary with time is because the transmitter and receiver are fundamentally asynchronous, and the actual received data rate generally will differ and vary with time from the expected data rate chosen to set the A/D and system clocks. Consider the design of Fig. 19; as part of the time-varying detection filter and slot-synchronization algorithm, the true slot clock is recovered. The true symbol clock also is recovered and is synchronous with the slot clock; however, both of these clocks are asynchronous with the A/D or system clock. Thus, the implementation of the architecture of Fig. 19 requires high-speed digital design with asynchronous clocks. The asynchronous nature of the processing also creates additional design challenges when a parallel or vector processing architecture is developed, as in the next section; these challenges are addressed in detail in Section VII.

Finally, of course it is not possible to achieve perfectly band-limited pulse shapes that are also full response or to implement perfect (infinite in time extent) interpolation filters, all of which was assumed for simplicity in the derivation of Fig. 19. However, these can be approximated very closely in reality, and generally such a system based on these principles results in negligible implementation losses if designed and implemented properly.

## VI. Parallel Discrete-Time Demodulator Architectures

Due to the very high sampling rates required by broadband optical pulses, processing rate reductions of 10 to 20 typically are required given the current state of the art if the targeted processing implementation is commercially available CMOS (FPGAs for example). Some demodulator functions are trivial to parallelize and others are not. The process of parallelization generally increases complexity; depending on the methods used, the increase in complexity may be linear or non-linear [8,9,14,15]. There are numerous methods, many based on frequency-domain or dual time-/frequency-domain processing, for parallelizing the convolution/correlation processing that reduce the complexity or transistor count of parallel implementations [9,14,15,37–40]. Figure 21 illustrates the most straightforward time-domain method of parallelizing the FIR filter structure illustrated in Fig. 20.

There is a much more elegant way of representing parallel filters and filter banks that is based on multirate notation. The vector downsample operation is given in Fig. 22. The parallel filter then can be represented by the structures given in Fig. 23, where the demultiplexer operation represents the delay and vector downsample chain (ordered serial-to-parallel conversion).

Using this notation, the parallel version of the demodulator given in Fig. 19 is given in Fig. 24. A core processing element of the demodulator is the time-varying parallel filter bank,  $F_{N,p}(z, u)$ ; this is also the processing element that may dominate the complexity (transistor) count of the VLSI implementation. There are numerous potential simplifications to the filter structure in Fig. 23—for example, if the impulse response is symmetric, the multipliers and adders may be reduced by a factor of 2. In addition, as indicated earlier, there are other options for performing the parallel filtering and many have greatly reduced multiplier requirements, depending on the filter type and order.

As mentioned earlier, one of the additional challenges of realizing the architecture in Fig. 19 or 24 in digital VLSI is appropriate asynchronous clock management. Additionally, the parallel implementation is much more straightforward to realize if the number of parallel paths (vector length) remains constant throughout a given processing operation (i.e., convolution). Obviously this creates challenges as each slot clock and symbol clock represents a fixed vector length of data samples and both of these are asynchronous with the A/D (system) clock. From a high level, the system of Fig. 24 is primarily composed of synchronous clocks and fixed-vector-length parallel processing. However, the data symbols must be clocked out at the true symbol rate that is asynchronous to the A/D clock and the  $1/M$ th rate system clock. One method of re-clocking the vector of symbols at the recovered data rate is with the asynchronous first-in first-out (FIFO) circuit, and there are many ways to realize this circuit. More details of the slot and symbol clock generation and the conceptual asynchronous FIFO-type circuits are presented in the next section.

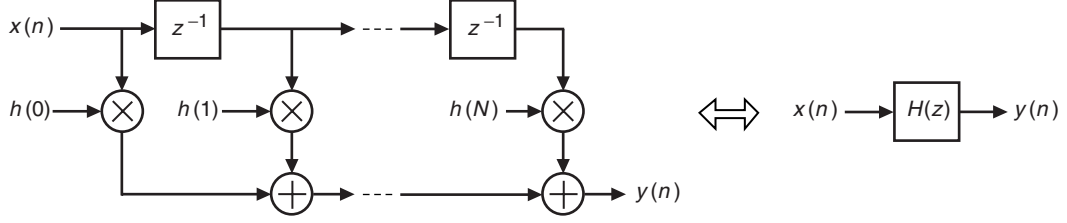


Fig. 20. FIR filter structure.

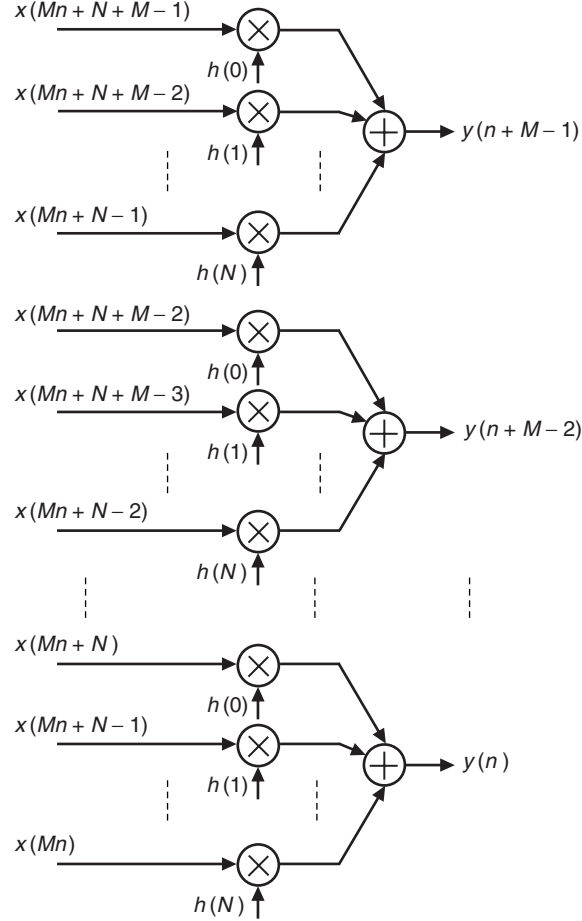


Fig. 21. Parallel FIR filter structure.

## VII. Asynchronous Discrete-Time Processing

The input signal has a dynamic nature due to the Doppler in the channel and absence of synchronization between the oscillators in the transmitter and receiver. As a result of this dynamic signal, the error  $\Delta \hat{t}_\varepsilon$  varies with time, and given the fixed-sample-clock A/D of Fig. 24, this results in a discrete-time system with asynchronous sample rates and a digital implementation with asynchronous clocks. The simplified error signal model may be represented as

$$\Delta t_\varepsilon(t) = T_{\text{slot}}(t) - T'_{\text{slot}}(t) + \tau \quad (13)$$

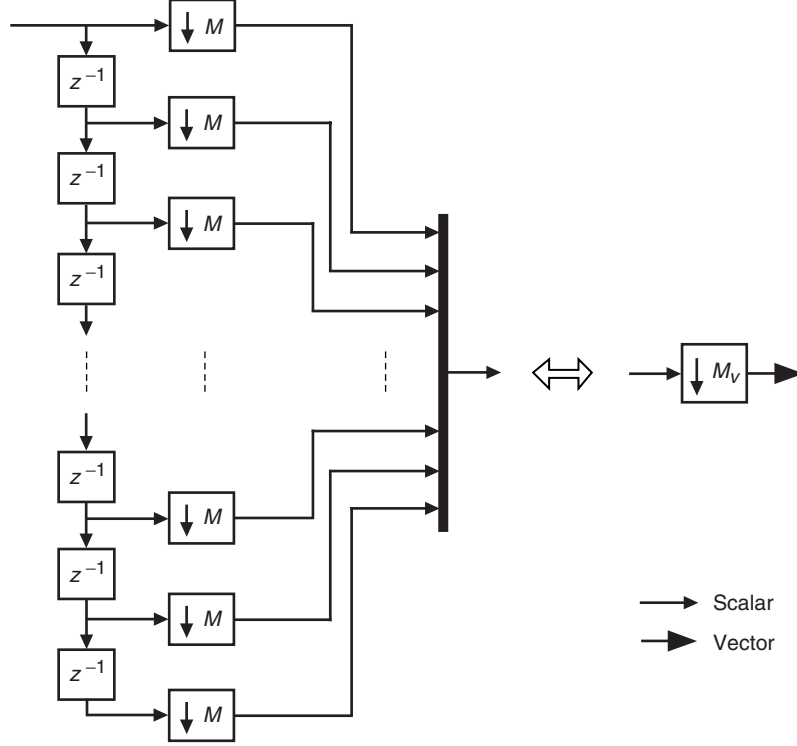


Fig. 22. Vector downsample operation.

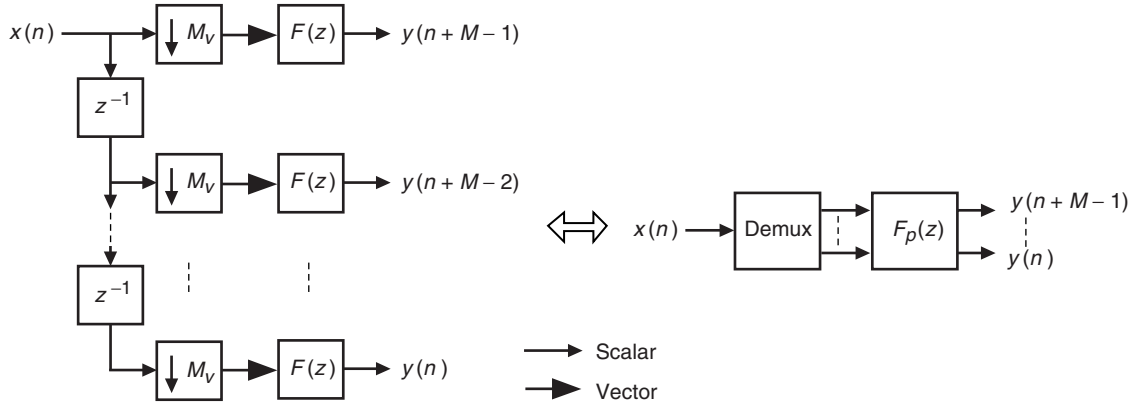


Fig. 23. Parallel filter bank representations.

where  $T'_{\text{slot}}$  is slot period due to Doppler, and  $\tau$  is a fixed time offset. The slot period with Doppler is given by

$$T'_{\text{slot}} = T_{\text{slot}} + T_{\text{slot}} \left( \frac{v}{c} \right) \quad (14)$$

where  $v$  is the relative velocity of the transmitter relative to the receiver, and  $c$  is the speed of light. Note that the relative velocity may be positive or negative. However, for reasons that will be demonstrated, significant simplifications result if the received sample clock frequency is set such that

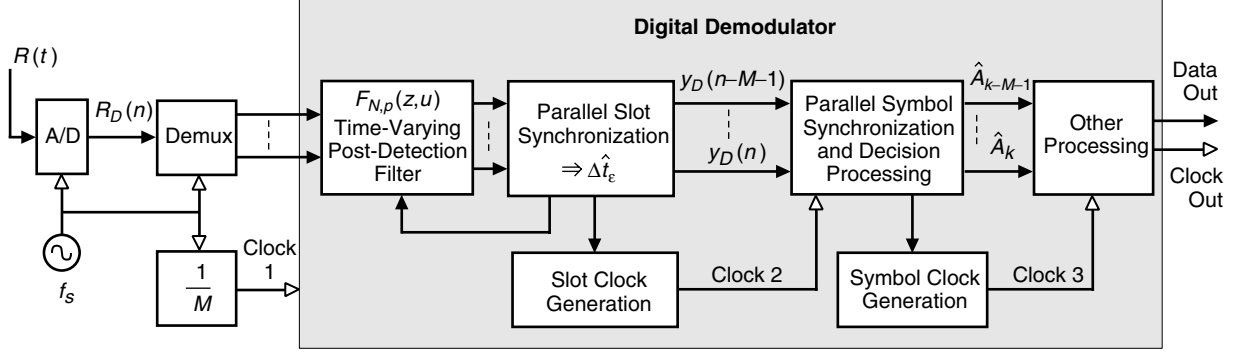


Fig. 24. Parallel discrete-time demodulator.

$$\frac{1}{T'_{\text{slot}}} \leq \frac{f_{S\_nominal} + r_1}{D} = \frac{f_S}{D} \quad (15)$$

The sample frequency is set such that the nominal sample rate assuming no Doppler,  $f_{S\_nominal}$  is increased by the real constant  $r_1$  such that the true sample frequency  $f_S$  divided by the designed number of samples per symbol is greater than the maximum slot frequency the demodulator processes. The slot-synchronization algorithm creates a new estimate at the nominal symbol rate,

$$\hat{\Delta t}_\varepsilon(nT_{sym}) \approx T_{\text{slot}}(nT_{sym}) - T'_{\text{slot}}(nT_{sym}) + \tau + \eta_0(n) \quad (16)$$

Here  $\eta_0(n)$  is some noise on the estimate. Successive timing-error estimates must be accumulated; this accumulation is represented as

$$\sum_n \Delta \hat{t}_\varepsilon(nT_{sym}) \approx \sum_n T_{\text{slot}}(nT_{sym}) - \sum_n T'_{\text{slot}}(nT_{sym}) + \tau + \sum_n \eta_0(n) \quad (17)$$

An accumulator such as this exists in the filter given in Fig. 9. As stated previously, the error estimate is an approximation to the difference between the nominal slot rate and the actual slot rate; this estimate is not generally equal to the actual error as many signal components and signal distortions are not included in this simple model. These include multiple jitter sources, channel and amplifier noise sources, imperfect filters, etc. Much analysis must be performed to determine actual performance of the estimator as a function of integration time or filtering and the statistical properties of the numerous random variables in the communications system. As shown in [2], this filtering may be accomplished with a closed-loop feedback system yielding very desirable Doppler tracking capabilities. Here we continue to address the implication of asynchronous processing on the overall architecture and ignore random noise and distortions.

The accumulated error may become arbitrarily large as  $n$  becomes arbitrarily large. For practical purposes, the error must be accumulated in a cyclic fashion, that is modulo some multiple of  $-T_s$ . Recall that the system clock rate of the serial demodulator in Fig. 19 is  $1/T_s$ . It is now demonstrated that it is desirable to alter the slot clock at time increments given by the following:

$$\text{round} \left\{ \sum_n \Delta \hat{t}_\varepsilon(nT_{sym}) \right\} = -\frac{L}{N} T_s \quad (18)$$

where  $L = kN$  and  $k$  is some integer greater than or equal to 1. For the demodulator in Fig. 19, there are clocks with periods  $T_s$ ,  $T_{\text{slot}}$ , and  $T_{sym}$ , as illustrated in Fig. 25.

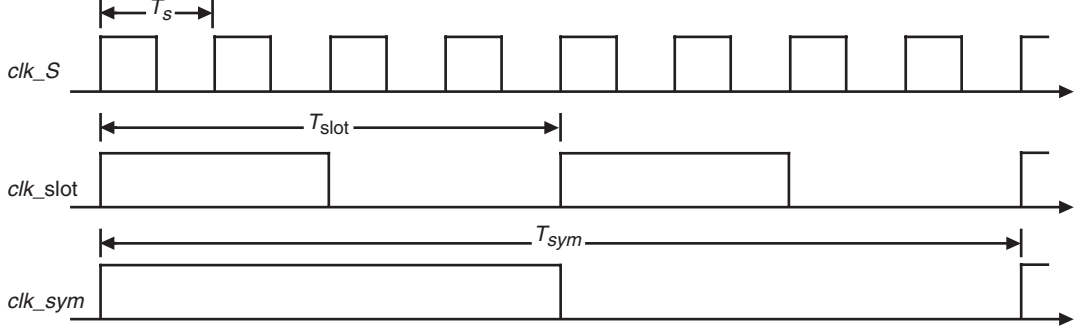


Fig. 25. Sample, slot, and symbol clocks.

Recall the slot period is  $D$  times the sample period,  $T_{\text{slot}} = DT_S$ . Here  $L/N = 1$ ,  $L/N = D$ , or  $L/N = DP$  in the accumulated error estimate represents the time of one sample clock, slot clock, or symbol clock cycle, respectively. The timing error in the signal is partially corrected with the filter bank  $F_{N,p}(z, u)$ ; the error is quantized to one of the time-shifted versions of the detection or interpolation filter. This error correction must be made modulo some time  $LT_S/N$ ; otherwise, arbitrary filter lengths and complexity are required if  $1/T'_{\text{slot}} < 1/T_{\text{slot}}$ , and the receiver runs for an arbitrarily long time. When the accumulated error time reaches its minimum value, it must wrap or roll over; otherwise, it will become arbitrarily large. Figure 26 illustrates the accumulation of the error estimate  $\Delta\hat{t}_\varepsilon$  modulo  $L/N T_S$ .

While the filter bank of Fig. 17 indicates a time delay of one sample period ( $N(T_S/N)$ ) from the first filter ( $u = 0$ ) to the last ( $u = N - 1$ ), it may be designed for any delay that is a ratio of  $L/N$  times ( $T_S$ ). It will be demonstrated that for certain parallel processing architectures specific delays result in much more simple overall implementations. To gain more insight into these issues, consider the example in Fig. 27:

- (1) Figure 27(a) is a received 2-PPM signal with 4 samples per slot ( $D = 4$ ) and no dynamics.
- (2) Figure 27(b) is a conceptual depiction of the effects of dynamics on the signal, where the amount of Doppler is greatly exaggerated to illustrate key concepts. In reality, timing error in Fig. 27(b) typically would be thousands or millions of times smaller per slot period.
- (3) Figure 27(c) demonstrates how the error is used to change  $u$ , thus choosing a filter with different subsample timing delay. For this example, assume that the slot filter is a perfect interpolation filter, so incrementing  $u$  creates a time-quantized delay, quantized to  $T_S/2$ , in the signal given in Fig. 27(b). Incrementing  $u$  by one is equivalent to incrementing one-half of one sample period. The maximum timing error due to time quantization with this filter bank is  $T_S/(2N) = T_S/4$ . The total delay in the filter bank is chosen to be  $-DT_S - T_S$ , which is one and one-quarter slot periods (recall there are  $D = 4$  samples per slot). To create a process that results in repetition of an entire symbol when the filter rolls over or wraps, the filter bank design method of Fig. 17 had to be extended by  $-4T_S$ .
- (4) In Fig. 27(d), when the timing error wraps it instantaneously moves the output signal back in time by  $4f_S$  (one slot period). The wrapping of the filter bank index  $u$  results in samples that already have been output by the filter being output again. These samples must be discarded, and the slot clock period associated with them also must be discarded. The period of the clock following the filter wrapping is instantaneously doubled; then the slot clock returns to its nominal period ( $T_{\text{slot}}$ ). This processing results in a slot clock with an average frequency that is equal to the recovered slot rate. In this way, the timing estimates are made and accumulated to adjust the clock period, which is readily accomplished in digital circuitry. There is obviously a need for a rollover detector to indicate when the

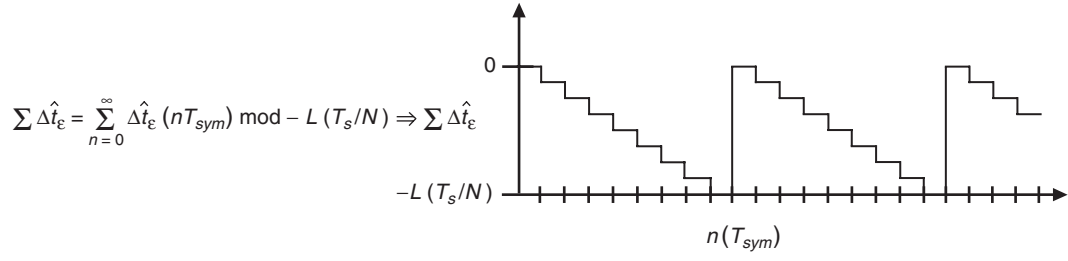


Fig. 26. Time delay.

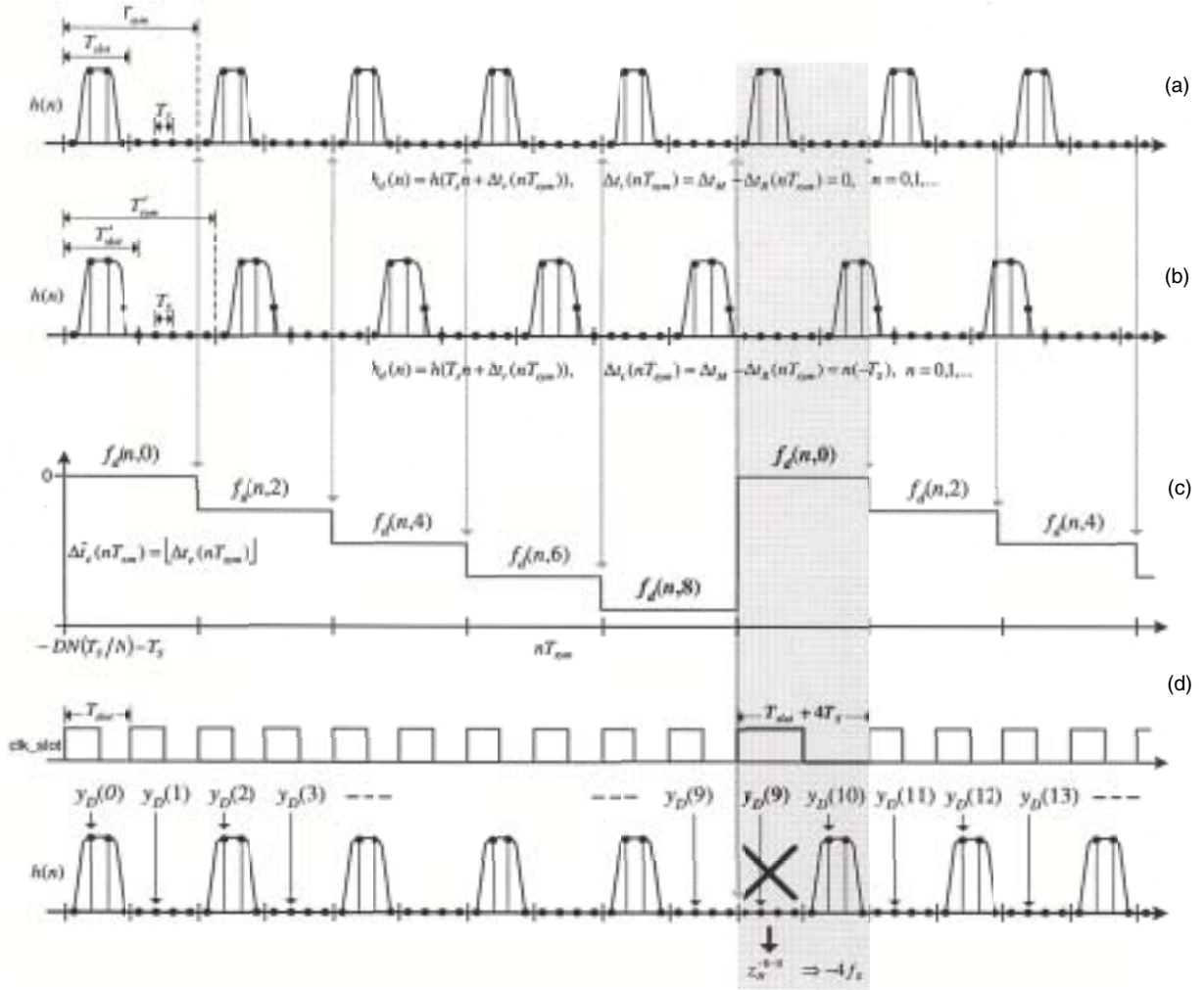


Fig. 27. Conceptual illustration of time-varying phase tracking and multirate processing.

clock period should be adjusted; this also is readily accomplished with digital circuits. This rollover detector may be designed to further average out or remove with thresholding the variations of the accumulated phase-error estimates due to noise and other distortions on the input signal, and thereby avoid multiple rollovers due to jitter on the error signal.

Although the memory length of the filter in Fig. 27 was chosen to be seemingly arbitrary to illustrate specific concepts, the number of samples that are repeated and then discarded after filter index rollover is a significant consideration in the design of the filter bank. It will be shown that it is very desirable to create the filter bank to roll over on slot boundaries, and this requires a modification to the earlier filter bank design method given in Fig. 17; consider Fig. 28.

In Fig. 28, the prototype filter,  $h_{dN}(n)$  or  $v_{dN}(n)$ , is zero padded; the filter is appended with a number of zero samples. The number of filters in the bank is now  $L$ ; setting  $L$  equal to an integer number of slots yields

$$\frac{L}{N}T_S = \frac{RDN}{N}T_S = RDT_S \Rightarrow L = RDN \quad (19)$$

In this case, the total time delay from the 0-indexed filter to the  $(L-1)$ -indexed filter is the number of samples per slot ( $D$ ) times the number of slots ( $R$ ) times the sample period, and the total number of filters is equal to  $RDN$ . A conceptual filter bank with  $R=1$  is illustrated in Fig. 29. The result is the ability to correct larger timing error before index rollover. This increased filter memory, however, results in more complex filter banks but allows control over the jump in time and corresponding number of samples discarded when the index to the filter bank rolls over. There is an equivalent multirate structure for the filter bank in Fig. 28 that results in significant simplifications, as illustrated in Fig. 30.

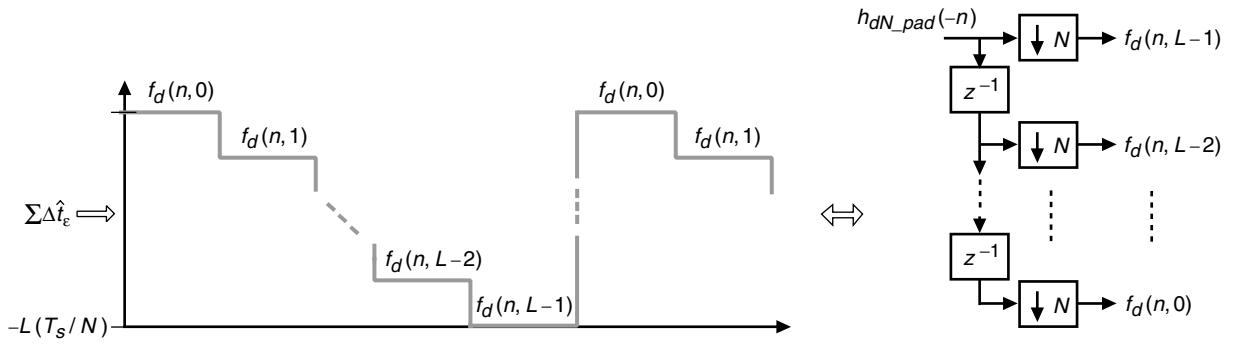


Fig. 28. Filter bank for correcting  $R$  slot times before rollover: prototype filter length  $L = R \times D \times N$ .

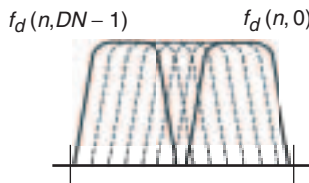


Fig. 29. Filter bank.

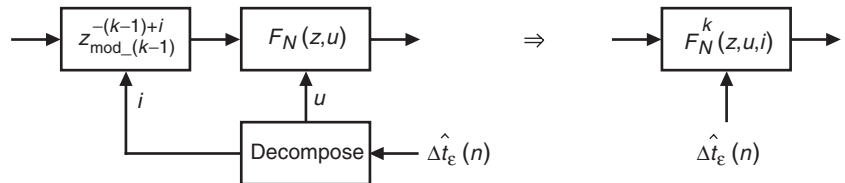


Fig. 30. Conceptual simplified filter bank.

The details of this simplified multirate filter bank are not presented here, but the structure incorporates a variable delay block before the filter bank. The delay is modulo  $k - 1$ , where  $L = kN$ . With this structure, the sub-sample errors are corrected by the filter bank while timing corrections that are integer multiples of a sample period are corrected by the variable delay. In this structure, the complexity of the filter bank is similar to that of Fig. 17 but increased by the addition of the variable delay. The error estimate  $\Delta\hat{t}_\varepsilon$  is decomposed into two components, one quantized to integer sample time error that is used to control the variable delay and another component quantized to sub-sample timing error used to control the filter bank. Note that if  $k = D$ , then the filter bank will roll over on slot boundaries.

The motivation for the filter index to roll over on slot boundaries is provided in Fig. 31. Consider the filter bank of the parallel architecture of Fig. 24, and note that after being processed by the filter bank the signal is input to the slot-synchronization algorithm. The current synchronization algorithms introduced earlier cannot have discontinuities in their input; if the signal is instantaneously shifted in time (in the discrete-time domain this is effectively possible), it must be done on a slot boundary or some modified slot-synchronization algorithm normalizing for the time shifts in the input must be developed. Note that the phase (timing error) wrapping can occur only across the vector boundary as this is when error estimates are updated in the parallel system. Error estimates are not updated within a vector block, only across vector boundaries, and when the rollover occurs, it is known precisely which data samples are repeated due to the rollover detector circuit; refer to Fig. 31.

There is another model of the time-varying filter bank that is useful for building further intuition useful for formal analysis beyond the scope of this article. The filter bank may be modeled as a numerically controlled oscillator (NCO) [16,38–40]. The variables  $u$  and  $i$  are updated every symbol and may be thought of as the error signal to the NCO. The phase of the NCO is changed by the error  $\Delta\hat{t}_R(nT_{sym})$ , and frequency is determined by the derivative of this estimate.

Figure 32 summarizes the results of this section from a high level. The asynchronous FIFO circuit is a critical subsystem of the demodulator. The functions of this circuit are to order the parallel vector appropriately for the parallel symbol synchronization and detection in the presence of the filter bank index

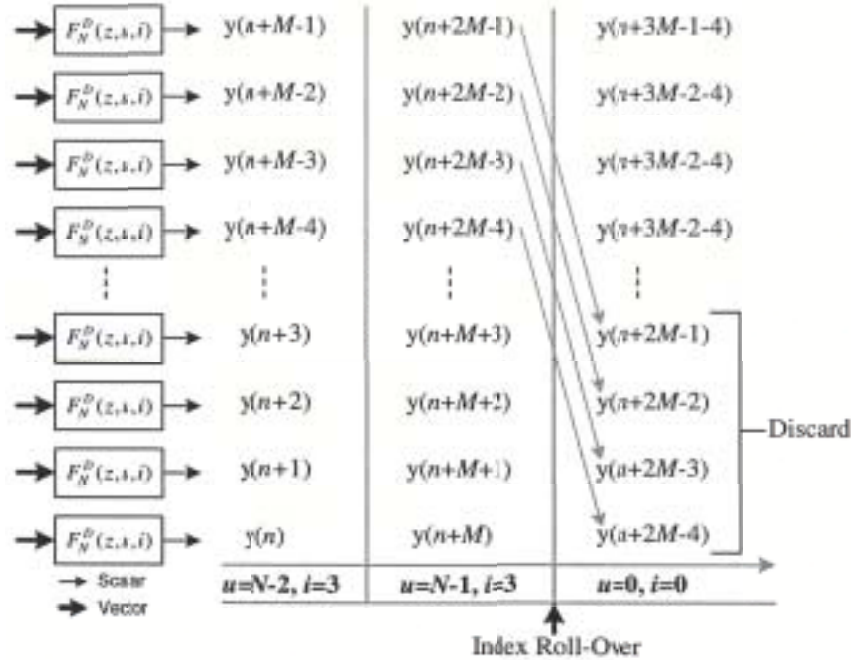


Fig. 31. Rollover on slot boundaries of parallel/vector system.



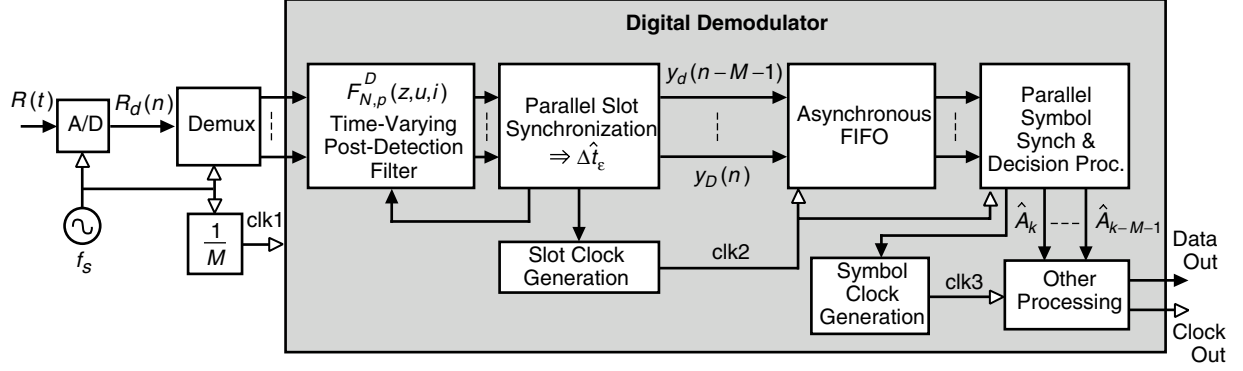


Fig. 32. Parallel digital demodulator.

rollover and to re-clock the data out at the appropriate recovered rate. The details of the asynchronous FIFO, such as memory depth, overflow, and underflow flags, and many other elements of this circuit are not presented here; there are many possible implementations to accomplish the functions of this circuit.

In conclusion, the elements of the architecture presented in this section are largely generic; they accommodate a time-variable post-detection filter whose purpose is twofold: filtering and the critical operation of correcting sample and sub-sample timing errors. This filter may be utilized with any number of slot-timing error-estimation algorithms and in a closed-loop system as in Figs. 7 through 9 and may approximate the high sample per slot A/D with continuous-time phase adjustments. It was inferred in this development and stated here explicitly that in addition to the challenges of designing the post-detection processing and slot-synchronization algorithms from a communications theoretical perspective, designing architectures for realizing these function in discrete-time at very high rates with Nyquist or near-Nyquist rate sampling entails many challenges. Overcoming these challenges requires developing and synthesizing discrete-time signal processing methods and algorithms for realizing the fundamental algorithms while overcoming the challenges of asynchronous and parallel digital design and implementation given the current state of the art in digital VLSI circuits. The discrete-time structures for realizing decision processing and symbol synchronization also pose many challenges. However, these challenges generally are more readily overcome than those for post-detection filtering and slot synchronization, namely sub-sample and asynchronous real-time processing at the maximum sample rate (A/D rate); as such, these architectures were not presented here in detail. With this as context, the next section develops more complete parallel discrete-time demodulator architectures.

## VIII. Parallel Discrete-Time Demodulator Architectures

### A. Simple Example Architecture

The purpose of this example is to illustrate further key considerations when synthesizing concepts developed earlier. A greatly simplified list of hypothetical requirements for a communication system and hardware limitations is given as follows:

- (1) Slot period,  $T_{\text{slot}} = 2$  ns (500-MHz bandwidth)
- (2) PPM pulse period = 1 ns (1-GHz bandwidth)
- (3) 4-64 PPM

Hardware limitations are as follows:

- (1) VLSI implementation in an FPGA with a maximum clock rate of 250 MHz
- (2) 4-Gigasamples-per-second A/D converter

Even from this short list of requirements, numerous design parameters may be determined. First the Nyquist sampling rate of the PPM pulse bandwidth and hence the system must be  $>1 \text{ GHz} \times 2 = 2 \text{ GHz}$ . It should be clear given the earlier discussions that the architecture development assumes  $D = 2k$ , where  $k$  is some integer. Although not stated explicitly, this assumption leads to multiple simplifications in the parallel-architecture development and facilitates many further simplifications in both the architecture and the hardware design that are beyond the scope of this work. There is also additional motivation for sampling above the Nyquist rate, design margin. Given that the A/D can accommodate a 4-GHz sample rate,  $D$  is set equal to 4. Given that the VLSI device implementing the demodulator architecture operates at 1/16th the rate of the A/D converter, the rate reduction or number of parallel paths  $M$  is set equal to 16.

Assume that some interpolation filter is used as the prototype filter in developing the filter bank  $F_N^4(z, u, i)$ . Determining  $N$ , the number of time-shifted filters in the filter bank, is generally not trivial. From a hardware complexity perspective,  $N$  should be made as small as possible to minimize complexity and transistor count. From a pure performance perspective,  $N$  should be made as large as possible to minimize residual sampling offset and synthesize as closely as possible the phase/time adjustments of the continuous-time system. Assume  $N = 128$ ; the residual sampling offset is then  $0.25 \text{ ns}/(2 \times 128) \cong 0.0009766 \text{ ns}$  or approximately 1 picosecond. This number may or may not be a good design choice depending on many other characteristics in the system both in the transmitter and in other subsystems of the receiver. For example, if the A/D oscillator phase jitter is significantly greater than 1 picosecond, it may not make sense to have the resolution of  $N = 128$ . Another perspective to consider is the performance, both tracking and acquisition, of the closed-loop tracking system of Figs. 7 through 9 (or other slot-synchronization algorithms) with time-quantized error correction. Much previous work exists in analyzing these and other issues; furthermore, a loop similar to that proposed has been analyzed, tested, and validated in hardware with  $N = 128$  [9]. However, detailed analysis of the performance of the type of loop with the optical channel has not been completed to date. The block diagram of a parallel architecture satisfying the simplified requirements given here is presented in Fig. 33 and includes a structure for a non-time-varying pulse equalizer  $B_1(z)$ .

Note that details of parallel symbol synchronization and decision processing are not given here, but they generally are significantly less complex to realize than are post-detection filtering and parallel slot synchronization. The challenging aspect of these former functions lies in the fact that they are parameterized on PPM order, but with the flexibility of a digital implementation such challenges are readily overcome. It should be noted that the number of parallel paths and the number of samples per slot in the slot-synchronization processing and filter  $F_N^D(z, u, i)$  are readily scalable by powers of two, and the amount of parallelization in the functions of symbol synchronization and decision processing is largely independent of these. Note that, due to the large overlap of vectors input to the vector downsample and parallel filter banks of Fig. 33, significant simplifications are possible in such parallel architectures; derivations of such simplifications are beyond the scope of this work.

As stated earlier, determining the performance analytically or pseudo-analytically of an architecture such as that in Fig. 33 for most realistic communications channels is a significant undertaking. However, given that lossless or perfect reconstruction parallelization techniques were used in its development, there is no need to determine the performance of this specific architecture. There is a serial equivalent architecture for which performance can be determined or estimated more readily, although this determination is still non-trivial with realistic channel models. This serial architecture is equivalent from a communications systems performance perspective, not from a processing-rate or complexity perspective that has been a focus of this work. The serial equivalent architecture is presented in Fig. 34. The delay  $Q_1$  is the total delay added by the parallel architecture in the feedback. For example, to model the actual delay in the implementation,  $Q_1$  must include the pipeline delays in all multipliers and adders in the feedback path.

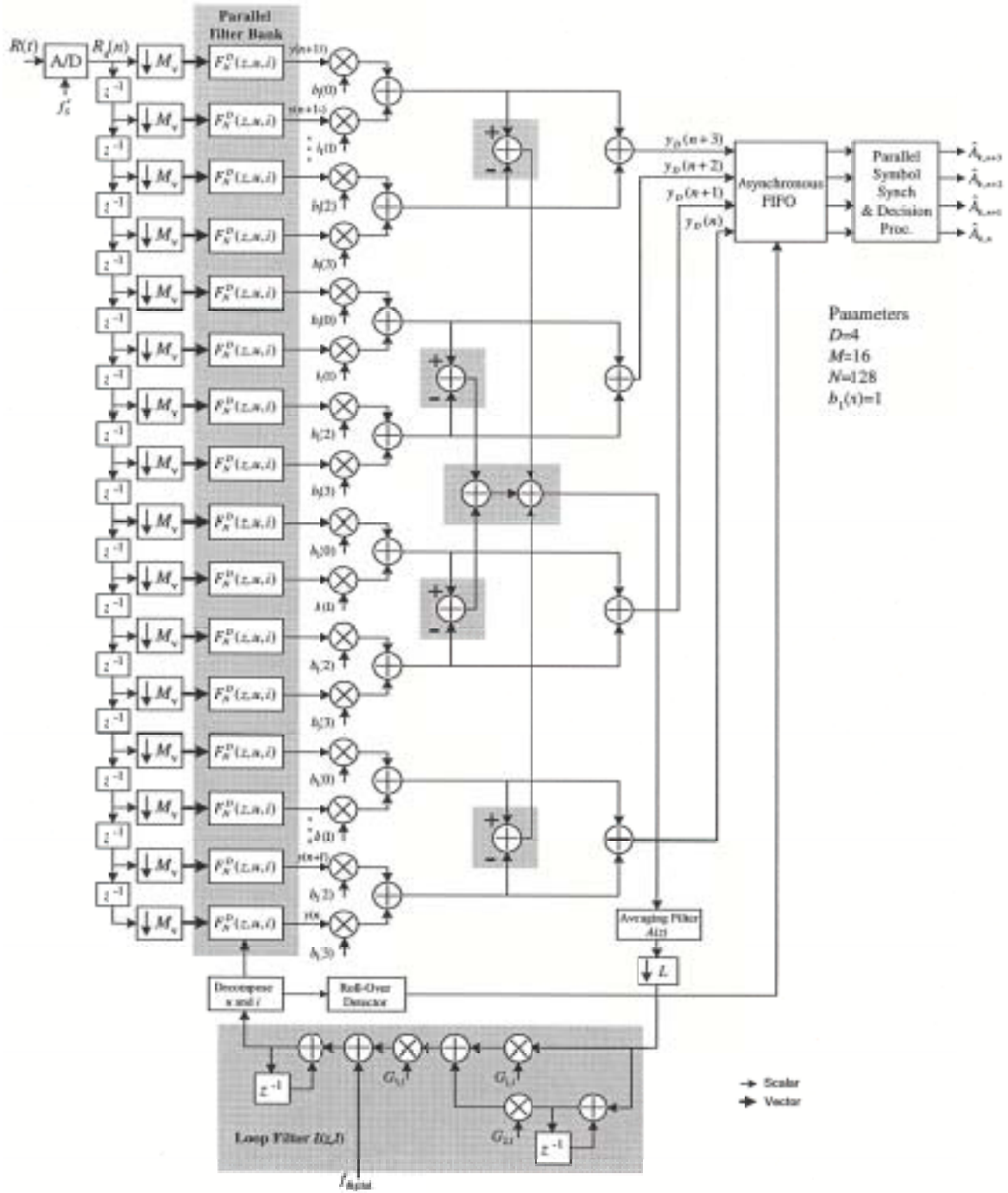


Fig. 33. 1/16th rate parallel architecture.

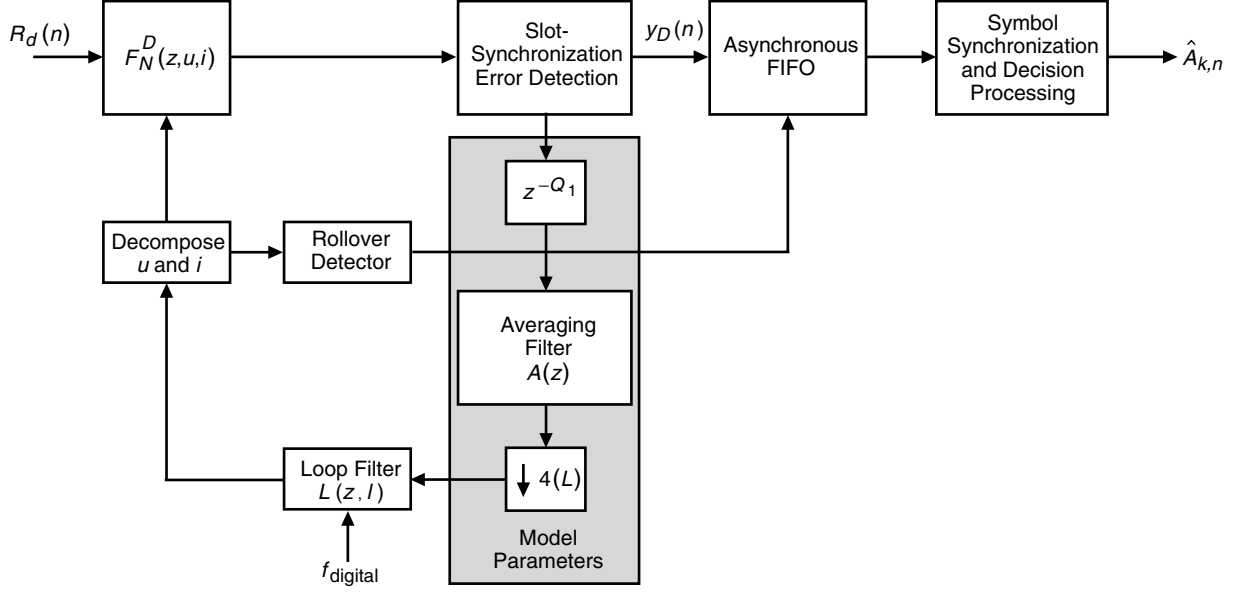


Fig. 34. Serial model of the parallel architecture.

## B. Performance with a Simple Optical Channel Model

A basic model of the optical PPM symbol detection uses photon-counting detection in the presence of background noise [1,2]. Thermal noise is negligible when using photon counting. This analysis assumes ideal slot and symbol synchronization. The number of photons in a coherent-state optical field generated by lasers is Poisson-distributed, i.e., the probability of detecting  $k$  photons,  $k \geq 0$ , is given by the expression

$$P(k) = \frac{K_s^k}{k!} e^{-K_s} \quad (20)$$

$K_s$  is the average number of signal photons per slot. The probability of correct detection for  $M$ -order PPM is

$$P_M(C) = 1 - \frac{M-1}{M} e^{-K_s} \quad (21)$$

The probability of symbol error is related to this quantity as

$$P_M(SE) = 1 - P_M(C) = \frac{M-1}{M} e^{-K_s} \quad (22)$$

When significant amounts of background light enter the receiver along with the signal, it is possible for the receiver to make an error even if one, or more, signal photons is detected, because a noise slot may occasionally produce a greater count than the signal slot. Let the average number of background photons be  $K_b$ . The probability of correct detection for maximum-likelihood detection of PPM symbols in the presence of noise has been derived in [1] and shown to be

$$P_M(C) = \left\{ \sum_{r=0}^{M-1} \left( \frac{1}{r+1} \right) \binom{M-1}{r} \sum_{k=1}^{\infty} \frac{(K_s + K_b)^k}{k!} e^{-(K_s + K_b)} \left[ \frac{(K_b)^k}{k!} e^{-K_b} \right]^r \left[ \sum_{j=0}^{k-1} \frac{(K_b)^j}{j!} e^{-K_b} \right]^{M-1-r} \right\} + M^{-1} e^{-(K_s + MK_b)} \quad (23)$$

The corresponding error probability is given by

$$P_M(E) = 1 - P_M(C) \quad (24)$$

Software simulations of the system in Fig. 34, with 4 PPM,  $N = 128$ ,  $D = 4$  samples/slot, and a normalized loop filter bandwidth of approximately 0.005 were performed using MATLAB<sup>®</sup> Simulink; a basic block diagram of the simulation model is presented in Fig. 35. The simulation assumed a fixed slot phase offset, thus requiring the slot- and symbol-synchronization loops to acquire and then track.

Using the mathematical model described in Eq. (1), the transmitted pulse,  $p(t)$ , was a perfect square pulse that occupied 50 percent of the slot period. No intensity variations other than those from the photodetector were modeled. The optical channel models only a simplified photon-counting type (possibly photomultiplier tube, PMT) of photodetector, as described above. The output of the channel is a Poisson distribution given by

$$y(k) = \frac{\Lambda^k}{k!} e^{-\Lambda}, \text{ with parameter } \Lambda = \begin{cases} K_s + K_b & \text{signal slot} \\ K_b & \text{no signal slot} \end{cases} \quad (25)$$

The software utilized a Poisson channel model, and the background mean photon count ( $K_b$ ) remained constant at 1 [2]. The signal mean photon count versus symbol-error rate (SER) is illustrated in Fig. 36. The simulation results plotted in Fig. 36 vary little from the theoretically predicted results given by Eq. (24); however, it should be emphasized that this is a simplified optical channel model.

### C. Evolved Parallel Architectures

The architecture of Fig. 33 is not a unique or final architecture design for the high-rate optical PPM demodulator; it is a starting point from which to evolve. A core processing element is the time-varying post-detection filter bank. This processing or some close variation is envisioned as being a key element of any all-digital demodulator; however, numerous simplifying signal processing structures for accomplishing this processing are possible. Other signal processing algorithms might be fundamentally improved or changed depending on the optical channel and advancing detection and estimation theoretical techniques derived for that channel. As an example, consider that the optimal PPM slot-synchronization signal processing algorithm currently has not been derived for the optical channel described in [2]. In addition to deriving such processing algorithms, improvements to existing synchronization algorithms might include a symbol-decision-directed slot-synchronization loop for the optical channel or pulse equalization. Indeed, the list of theoretical improvements or channel-specific designs for synchronization, post-detection

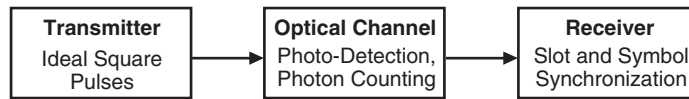
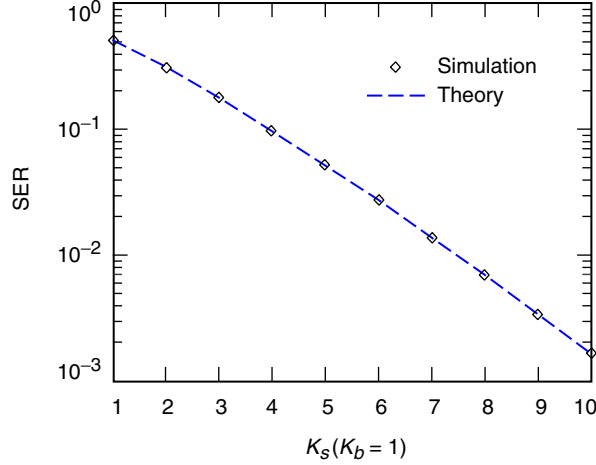


Fig. 35. Software model block diagram.



**Fig. 36. Performance of the software model.**

filtering, and decision processing, as well as other processing subsystems, is quite long. In addition, as the state-of-the-art hardware implementation devices improve, more computationally complex signal processing may be feasibly implemented. Many future algorithm improvements are currently not possible to predict with a high degree of accuracy. However, there is a set of architecture evolutions or improvements that are largely possible to predict. These are the signal processing functions that, while not the focus of this work, are a necessary part of a ground receiver or demodulator. The demodulator architecture will evolve to contain these functions:

- (1) Master controller (for real-time autonomous demodulator monitoring and control)
- (2) Forward error-correction preprocessing
- (3) Power estimation and SNR estimation
- (4) Slot-synchronization and symbol-synchronization lock detection
- (5) Digital automatic gain control

Figure 37 illustrates an evolved parallel demodulator architecture incorporating additional functions. Many additional functions have been added to the baseline architecture of Fig. 33, and the slot-synchronization loop incorporates the option of being decision directed (note the switches). The baseline parallel processing and sub-sample timing-error correction of Fig. 33 remains intact. Examples of other architecture improvements and alternatives are presented next. We will demonstrate motivation for the filter bank for timing-error estimation, to be designed independently of the post-detection filter bank used for filtering and timing-error correction on the signal used for decision processing; the process of synchronization and post-detection filtering may be separated or deconstructed. It will become clear that there are several advantages to deconstructing the processing this way. Part of the motivation for this is that different filter bank designs and processing rates may be used for the synchronization and post-detection filter, and this may facilitate further advantages in the design and implementation of the synchronization filters.

Consider the architecture in Fig. 38; this architecture may be designed to be mathematically equivalent to the slot-synchronization architecture given in Fig. 33.

Consider the architecture in Fig. 39; this architecture is a straightforward modification to that of Fig. 38 and incorporates a simple averaging filter with all-ones coefficients that is downsampled by a rate that is equal to the number of coefficients that are in the filter ( $L$ ). Very large filter orders are

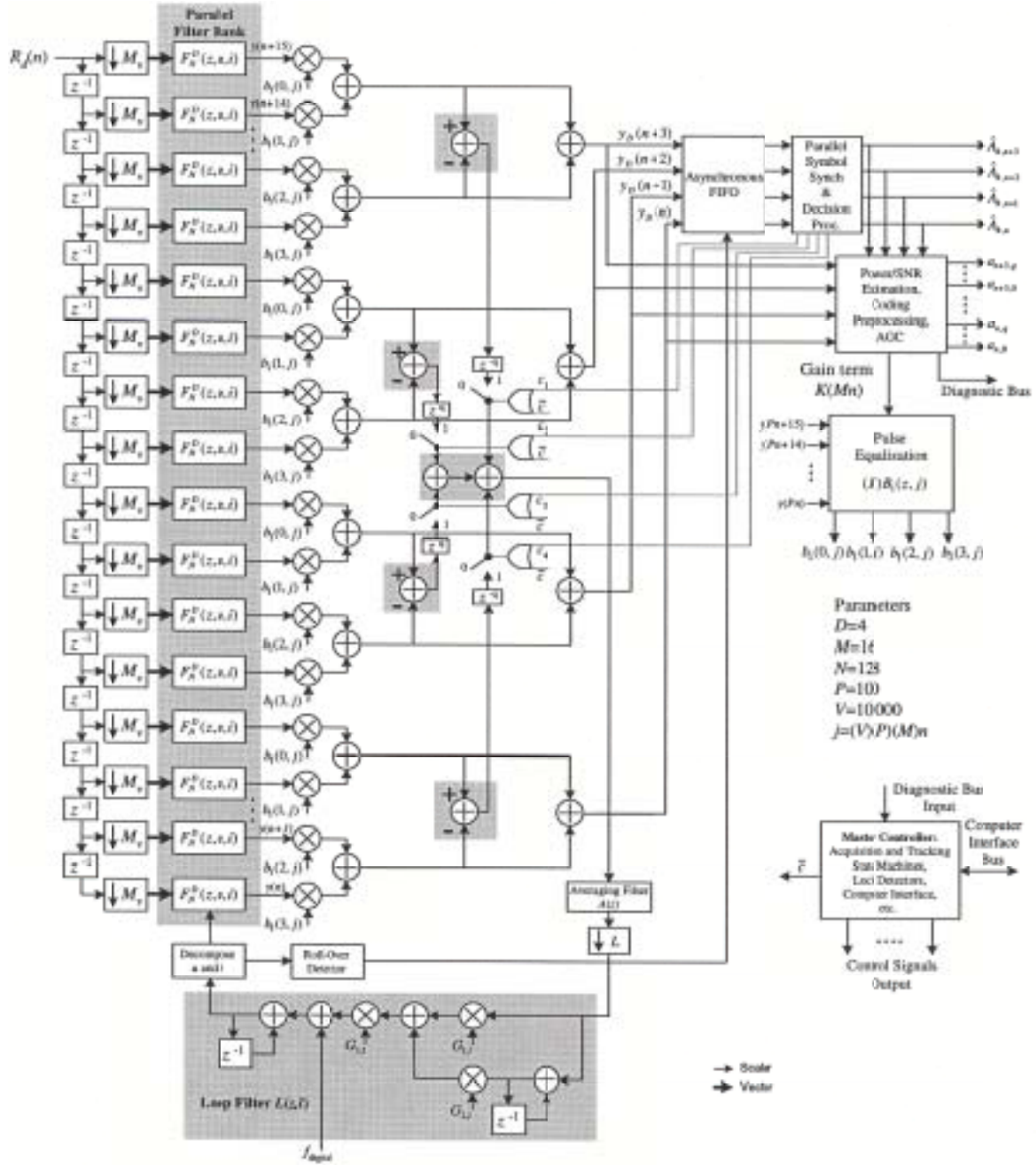


Fig. 37. A more comprehensive architecture; core processing of Fig. 33 retained.

possible with the simple feedback structure as illustrated in the figure. The simple averaging filter depicted is often referred to as the integrate-and-dump filter. The input to this averaging filter may be switched with control signals  $c_i$  to be either zero ( $c_i = 0$ ) or the sampled input signal ( $c_i = 1$ ). This is done to allow feedback control from the decision processing such that only samples estimated to contain a PPM pulse are input to the slot-synchronization algorithm. Note that, due to the large overlap of input vectors in Figs. 39 and 33, significant further simplification is possible in the realization of these architectures.

Each output line of the vector downsample block in Fig. 39 is a vector, so there is actually a vector of switches and averaging filters operating at rate  $f_s$ , but the output of these filters is clocked at rate



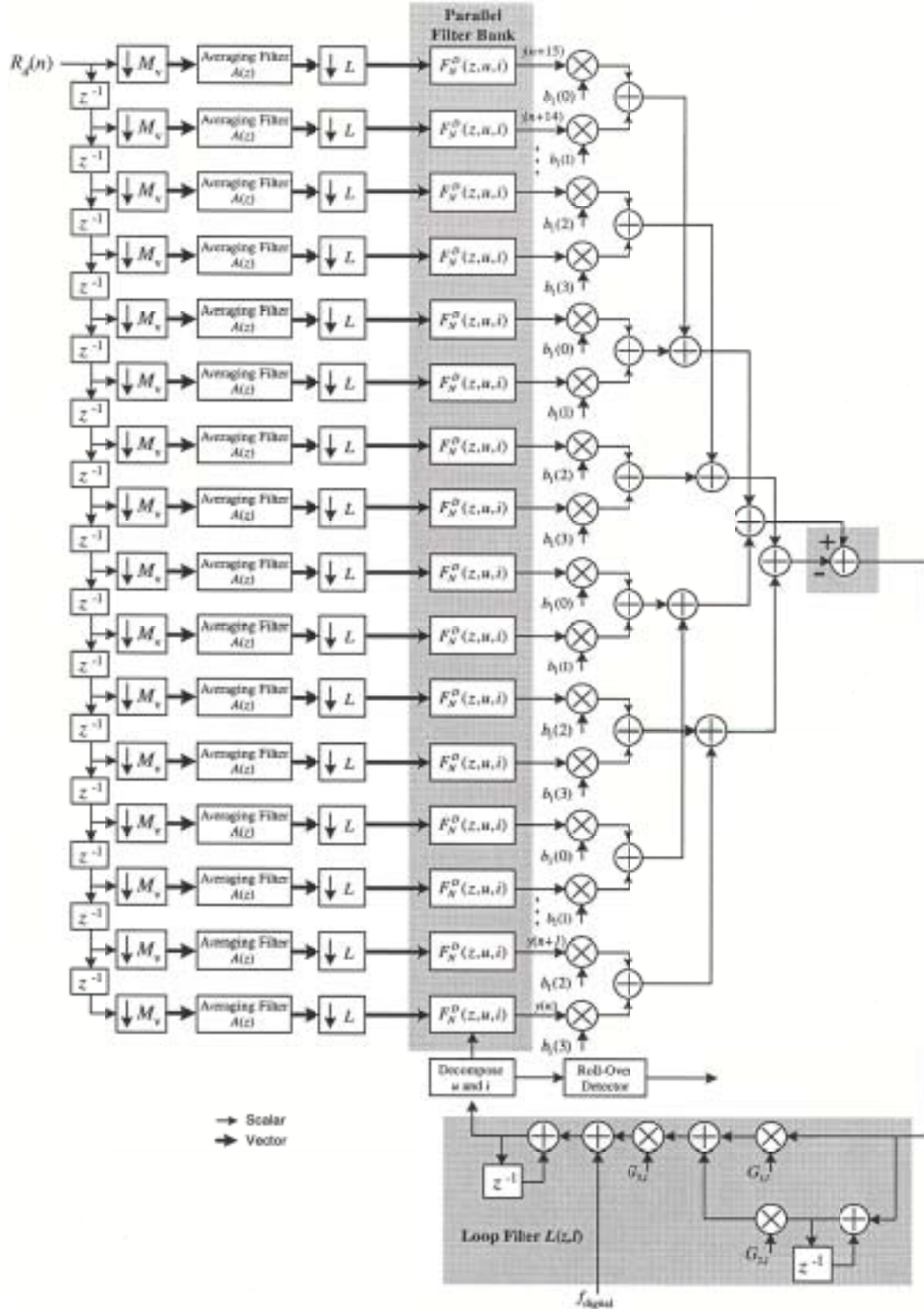
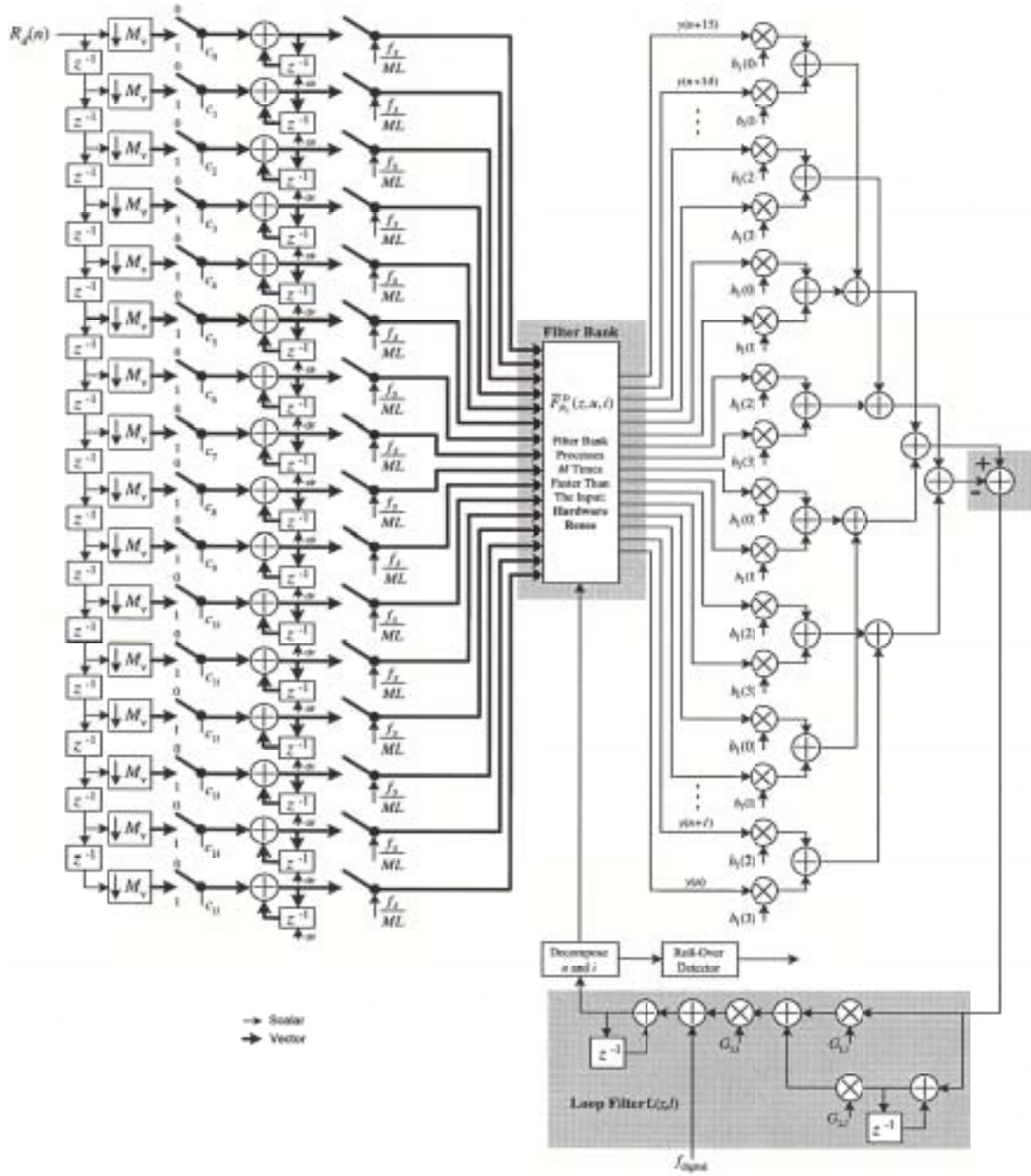


Fig. 38. Alternative slot-synchronization architecture equivalent to that of Fig. 37.





**Fig. 39. Simplified alternative slot-synchronization architecture; filter rate =  $M \times$  (the parallel input rate).**

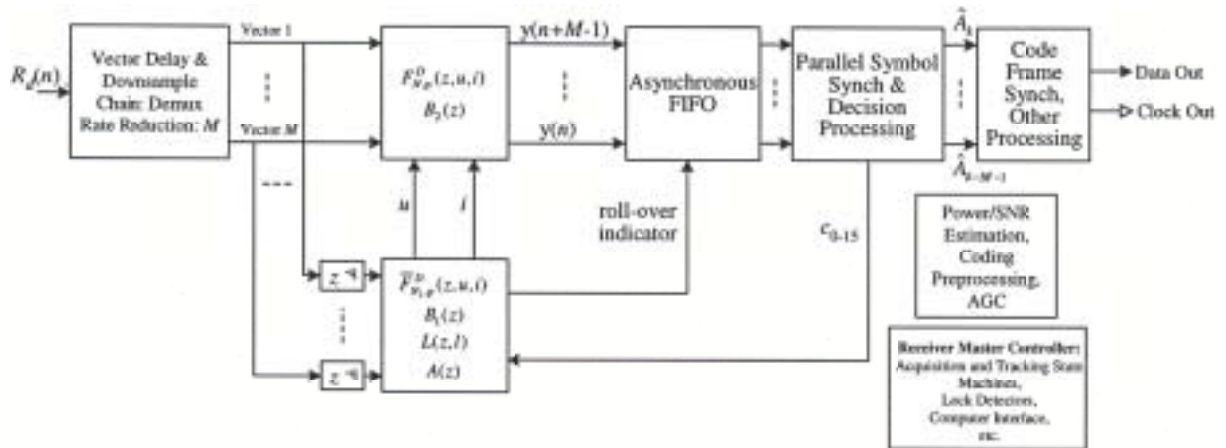
$f_S/(ML)$ ; this is the input rate to the time-varying filter bank. However, the filter bank  $\bar{F}_{N_1,p}^D(z, u, i)$  operates at rate  $Mf_S/L$ , that is,  $M$  times faster than its input, because this one filter structure is used to process all inputs. This is referred to as hardware reuse. This architecture requires 1/16th the number of concurrent multipliers of that required by Fig. 38 for the same prototype filter order (same signal processing performance). This may result in a significant savings in concurrent multipliers but with some added complexity of register delays and other controlling logic that are required in the hardware reuse architecture. This complexity reduction may result in a filter bank design  $\bar{F}_{N_1,p}^D(z, u, i)$  with more multipliers and/or more time-shifted filters (larger  $N$ ) than otherwise would be feasible to

implement. *This complexity reduction increases the design options of the filter  $\bar{F}_{N_1,p}^D(z, u, i)$  and, thus, the design and performance options of the slot-synchronization algorithm that are feasible.* It should be emphasized that a full-rate filter bank  $F_{N,p}^D(z, u, i)$  is still required to filter and correct the timing error of the input signal that is then input to the decision process (recall this is the process of estimating the transmitted symbol). The architecture with deconstructed slot-synchronization (timing-error estimation) and timing-error correction in the post-detection filtering with some pulse equalizer  $B_2(z)$  is illustrated in Fig. 40.

The time-varying filter  $\bar{F}_{N_1,p}^D(z, u, i)$  and pulse equalizer  $B_1(z)$  for slot-timing error estimation may be designed and implemented independently of the time-varying post-detection filter  $F_{N,p}^D(z, u, i)$  for filtering and signal-timing error correction and of the equalizer  $B_2(z)$  for pulse filtering (possibly for pulse-distortion mitigation). Note the signal delay ( $z^{-q}$ ) between the slot-synchronization processing and the signal path containing the post-detection filter bank and the decision processing. This is to facilitate appropriate feedback from symbol synchronization to slot synchronization if such a decision-directed system is desired. This delay is not in the feedback path of the closed-loop synchronization algorithm. This is not the case for the architecture of Fig. 37, which required a delay in the feedback path in order to be decision-directed; this delay may be small, but any delay in the feedback has the potential to degrade synchronization performance.

To summarize the advantages of the architecture in Fig. 40, first filter bank  $\bar{F}_{N_1,p}^D(z, u, i)$  may be designed and optimized for synchronization estimation performance independently of the filter bank  $F_{N,p}^D(z, u, i)$  used for correction of timing errors and filtering for the decision process. The filter bank  $\bar{F}_{N_1,p}^D(z, u, i)$  may incorporate many more design choices and may be many times more computationally intensive, that is, a larger number of coefficients, and/or larger resolution, and/or larger  $N$  than would otherwise be possible. The architecture of Fig. 40 does not require an additional delay in the synchronization feedback path in order to be decision-directed. This architecture may or may not be a good choice over the architectures of Fig. 33 or 37. Depending on the system, desired performance and regions of operation, and the resulting filter bank design(s), the architecture of Fig. 40 may be more or less computationally intensive than the architectures of Fig. 33 or 37.

The filter bank  $F_{N,p}^D(z, u, i)$  may be designed for specific pulse shapes and jitter conditions. Although the on-chip memory requirements likely preclude one of numerous banks of filters from being chosen on the fly without reconfiguration, there is another solution. As an example of the flexibility of this



**Fig. 40. Alternative parallel demodulator architecture: slot-synchronization error estimation with filter bank  $\bar{F}_{N_1,p}^D(z, u, i)$  and equalizer filter  $B_1(z)$  separated from the post-detection filter bank  $F_{N,p}^D(z, u, i)$  and detection equalizer  $B_2(z)$ ; the processing rate of the filter  $\bar{F}_{N_1,p}^D(z, u, i) = M(f_s)/L$  and filter  $F_{N,p}^D(z, u, i) = f_s/M$ .**

architecture when targeting reconfigurable (FPGA) VLSI implementation multiple filter banks,  $k$  filter banks,  $F_{k,N,p}^D(z, u, i)$ , may be developed and stored off-line and reconfigured for field tests and experiments very rapidly (seconds or less).

Finally, the architectures presented in this section are examples of those that may be developed given the methods and framework presented in this work along with other multirate discrete-time signal processing theorems and communications demodulator algorithms found in the literature. Many further discrete-time PPM demodulator architecture variations may be developed with varying degrees of complexity and performance. The conclusion includes a discussion of the high-speed digital platform suitable for implementing such architectures and the development process that facilitates an evolving architecture development incorporating software modeling and simulation, analysis, laboratory experimentation, and rapid prototyping. We now present the key design equations for understanding the architecture from a frequency (clock rate) and data throughput perspective; these are useful in understanding the complexity and challenges of the high-speed platform that is used to implement the class of architectures presented here.

## IX. Primary System Models and Parameters

There are several aspects of the discrete-time architecture illustrated in Fig. 33 that must be considered in the development of the high-speed digital platform; two of the most significant are the frequency (clock rate) requirements and the number of required data transmission lines. These requirements are largely determined by the sample rate of the system along with the hardware limitations of the digital devices used to realize the discrete-time or digital architectures that together determine  $M$  and the number of parallel input/output signals.

The primary clock frequencies are represented in Figs. 41(a) through 41(d); the maximum frequencies are in the right-most shaded column. These clock relationships are critical for such things as high-speed platform design and VLSI circuit design and floor planning. The relationship between the clock frequency on a given axis and the clock frequency on one axis above is given in the left-most shaded column. The fastest digital clock rate is the sample frequency (also referred to as the system frequency) on the top axis; this rate is reduced by the serial-to-parallel conversion, resulting in the divide by  $M$ . The slot rate clock is represented on the second axis; the symbol frequency is given on the third axis, and the bit clock frequency is given in the fourth axis. The slot-synchronization loop bandwidth is illustrated on axis as a fraction of the symbol rate.

Certain fundamental design equations of the architecture given in Fig. 33 are now summarized. The variables in these equations fall into two categories: design parameters and operational parameters. It is a key step in the development of complex VLSI systems to assign all parameters to one of these two categories. Such categorization is essential for establishing priorities and the associated schedule for establishing parameter values. This in turn allows development resources to be focused appropriately for various development strategies, one of which is summarized in the conclusion of this work. The two categories are defined as follows:

- (1) Design parameters determine the physical construction of the prototype board layout and design, the VLSI device, package, and numerous other significant elements of the design. These physical parameters generally are either not reconfigurable after manufacture or require significant redesign. These parameters should be determined before significant design work is undertaken.

- (2) Operational parameters may take on a variety of values depending on the requirements of the system. The operational parameters are either programmable or reconfigurable without significant redesign. The values of these parameters generally do not need to be known precisely before design or even during hardware implementation and test. However, the range, the maximum and minimum values, should be established or estimated as early as possible in the development process.

The top-level design equations and parameters for the architecture in Fig. 33 are given in Table 1. This is not a comprehensive list of equations describing the digital demodulator, but the most essential ones in early-phase hardware development.

Figure 42 illustrates a simple data I/O model of the architecture in Fig. 33. This model indicates the approximate number of digital input/output transmission lines required between major signal processing subsystems of the demodulator. Of primary concern is the number of transmission lines at the output of the serial-to-parallel converter. The number of lines cannot exceed input capability of the device used to implement the rest of the demodulator. At the same time, the number of lines must support the rate reduction required.

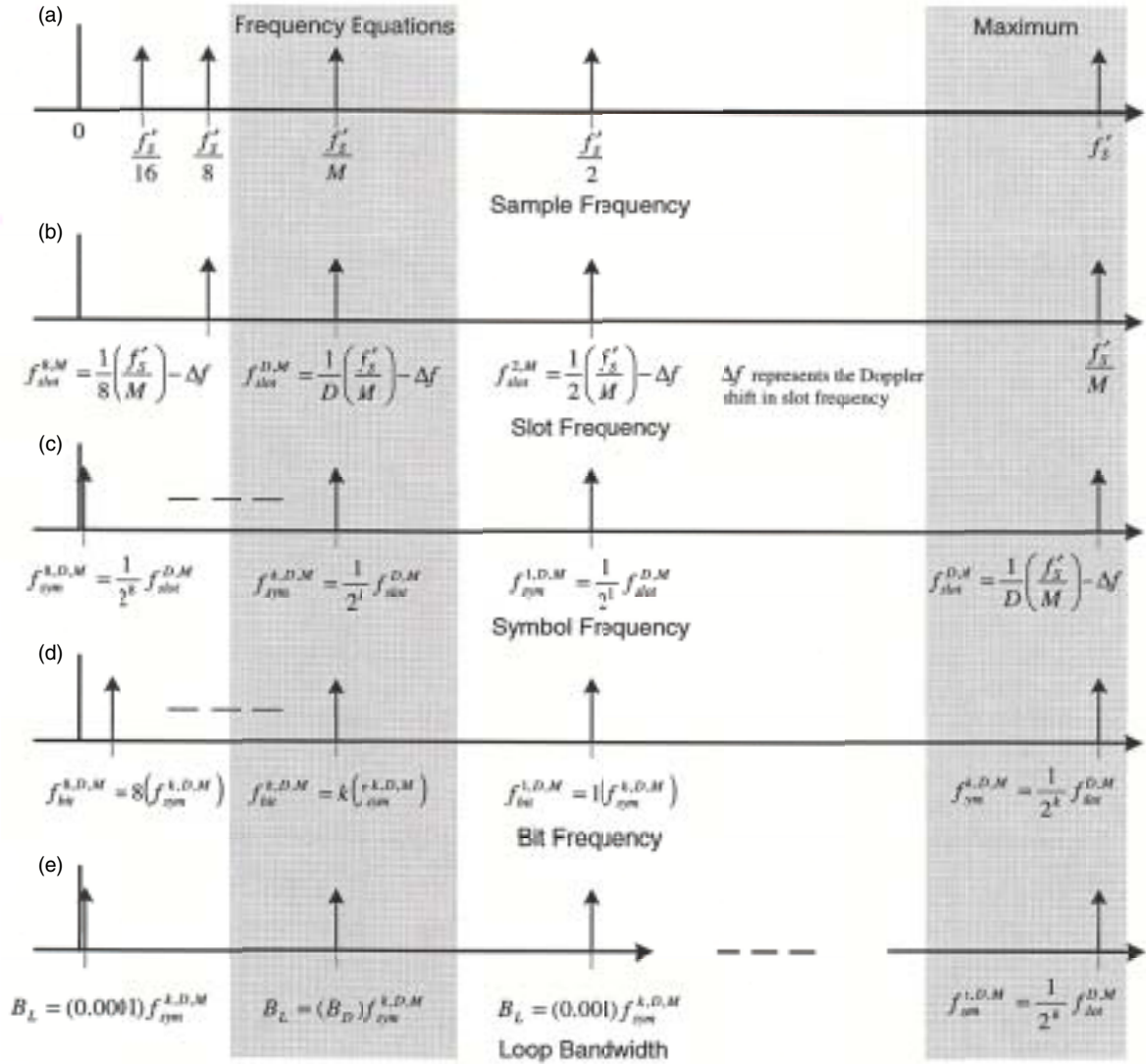


Fig. 41. Frequency model.

Table. 1. List and description of primary design functions.

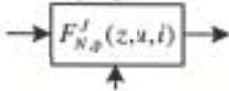
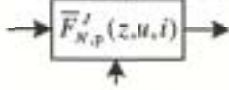
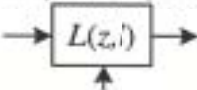
Design Function	Parameter Description
$f_{\text{bit}}^{k,D,M} = \frac{k}{2^k DM} \left( \frac{f_s}{M} \right) = \underbrace{\left( \frac{k}{2^k} \right)}_{\text{Operational Parameter}} \underbrace{\left( f_s \right) \left( \frac{1}{D} \right) \left( \frac{1}{M} \right)}_{\text{Design Parameters}}$	<p><i>Operational Parameter:</i>  <math>k</math> = The number of bits per symbol</p> <p><i>Design Parameters:</i>  <math>D</math> = Nominal number of samples per slot  <math>M</math> = Processing rate reduction  <math>f_s</math> = Ideal sample rate, not actual sample rate</p>
$f_{\text{sym}}^{k,D,M} = \frac{1}{2^k} f_{\text{slot}}^{D,M}$	<p><i>Operational Parameter:</i>  <math>k</math> = bits per symbol  <math>2^k</math> = Number of slots per symbol</p>
$\frac{D}{T_{\text{slot}} + T_{\text{slot}} \left( \frac{v_{\text{min}}}{c} \right)} \leq f'_s$	<p><i>Design Parameter:</i>  <math>f'_s</math> = The actual sample rate must be greater than the Doppler shifted slot rate times <math>D</math></p>
<p>Parallel Time-Varying Post-Detection Filter</p>  <p><math>i</math> = sample index  <math>u</math> = sub-sample index  <math>h_{\text{dN}}(n)</math> or <math>v(n)</math> prototype filter</p>	<p><i>Operational Parameters:</i>  <math>N</math> = Prototype FIR filter oversample rate, maximum residual sample offset = <math>T_g/(2N)</math>, there are <math>N</math> filters in the filter bank  <math>RDH</math> = Number of coefficients in the prototype filter  <math>J(T_g)</math> = Timing error accumulated before roll-over</p>
<p>Parallel Time-Varying Slot-Synch Filter</p>  <p><math>i</math> = sample index  <math>u</math> = sub-sample index  <math>h_{\text{dN}}(n)</math> or <math>v(n)</math> prototype filter</p>	<p><i>Operational Parameters:</i>  <math>N</math> = Prototype FIR filter oversample rate, maximum residual sample offset = <math>T_g/(2N)</math>, there are <math>N</math> filters in the filter bank  <math>RDH</math> = Number of coefficients in the prototype filter  <math>J(T_g)</math> = Timing error accumulated before roll-over</p>
<p>Slot Synch Loop Filter</p>  <p><math>l</math> = filter tracking/acquisition index  <math>B_{LD} = (B_N) f_{\text{sym}}^{k,D,M}</math></p>	<p><i>Operational Parameters:</i>  <math>B_{LD}</math> = Discrete-time bandwidth of IIR loop filter: <math>G_{1,l}, G_{2,l}, G_{3,l}</math>  <math>B_N</math> = Discrete-time bandwidth normalized as a fraction of the symbol rate  <math>(B_L = \text{Prototype analog loop filter bandwidth})</math></p>
$\frac{1}{T_{\text{slot}}} \leq f_{\text{digit}} \leq \frac{1}{T_{\text{slot}} + T_{\text{slot}} \left( \frac{v_{\text{min}}}{c} \right)}$	<p><i>Operation Parameter:</i>  <math>f_{\text{digit}}</math> = Normalized digital slot frequency rate</p>

Table. 1. List and description of primary design functions.

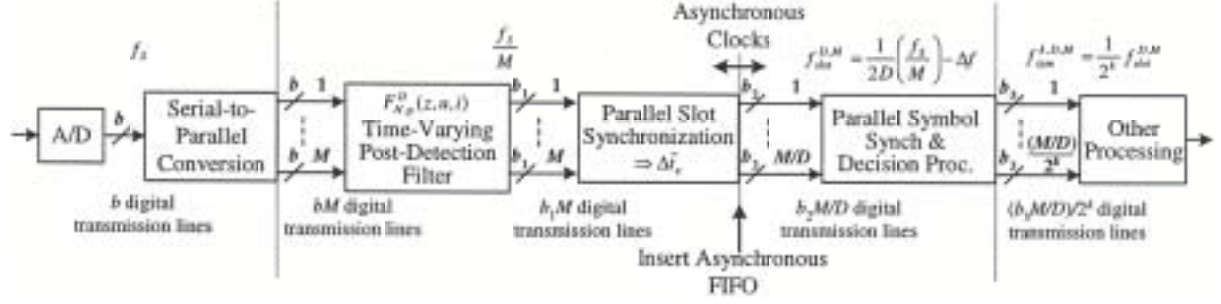


Fig. 42. Data I/O model.

## X. Conclusion and Future Work

Discrete-time demodulator architectures for broadband free-space optical PPM that are capable of Nyquist or near-Nyquist data rates were presented. The architectures were developed within a framework that encompasses a large body of theoretical work in optical communications, synchronization, and multirate discrete-time signal processing and were constrained by many of the limitations of the state-of-the-art digital hardware. The primary focus of this work was on the development of discrete-time processing of the most fundamental algorithms and processing required in PPM demodulators—those necessary for post-detection filtering, synchronization, and decision processing. Numerous other processing subsystems are required in an operational receiver. Integrated in the development and discussion were numerous considerations of modern VLSI devices as well as platform design. The most fundamental design decisions, those globally impacting platform design or fundamentally limiting performance, are summarized as follows:

- (1) The development targets an all-digital approach as opposed to a hybrid analog–digital or primarily analog implementation.
- (2) The all-digital design or lack of feedback from digital-to-analog processing results in tremendous flexibility without redesign; the many orders of magnitude possible (typically greater than 8 with current CMOS) with digital implementation can be achieved without the difficulties and limitations of the digital-to-analog feedback approach.
- (3) The fixed-sample-per-slot design results in a simplified demodulator development. As the symbol rate changes, the number of samples per slot remains constant. Once one data rate has been tested and verified, a large subset of characteristics and parameters for all data rates has been verified; the end result is a more simple and time-efficient development. The downside of this approach is that the anti-aliasing filter before the A/D may require a variable bandwidth; these issues are discussed further in [8,9]. This latter is one motivation for not using this fixed number of samples per slot approach, and there are others. It should be noted that the design process and reconfigurable platform do not fundamentally limit the choice of alternative designs for different data rates or a different number of samples per slot. With an FPGA approach and appropriately designed platform, the decision to use a different number of samples per slot may be made at virtually any phase of the development, although architecture iteration would be required.

There are numerous other design decisions that were made here and even more that are yet to be made during the development. Many of these decisions require further analysis, such as precisely which slot-synchronization algorithm to use and its parameter values for various regions of operation (SNR and signal dynamics), and many are decisions made by digital designers that have no direct impact on communications performance and are not dependent on regions of operation. It must be understood that a great many of these open design questions cannot be answered by considering and optimizing



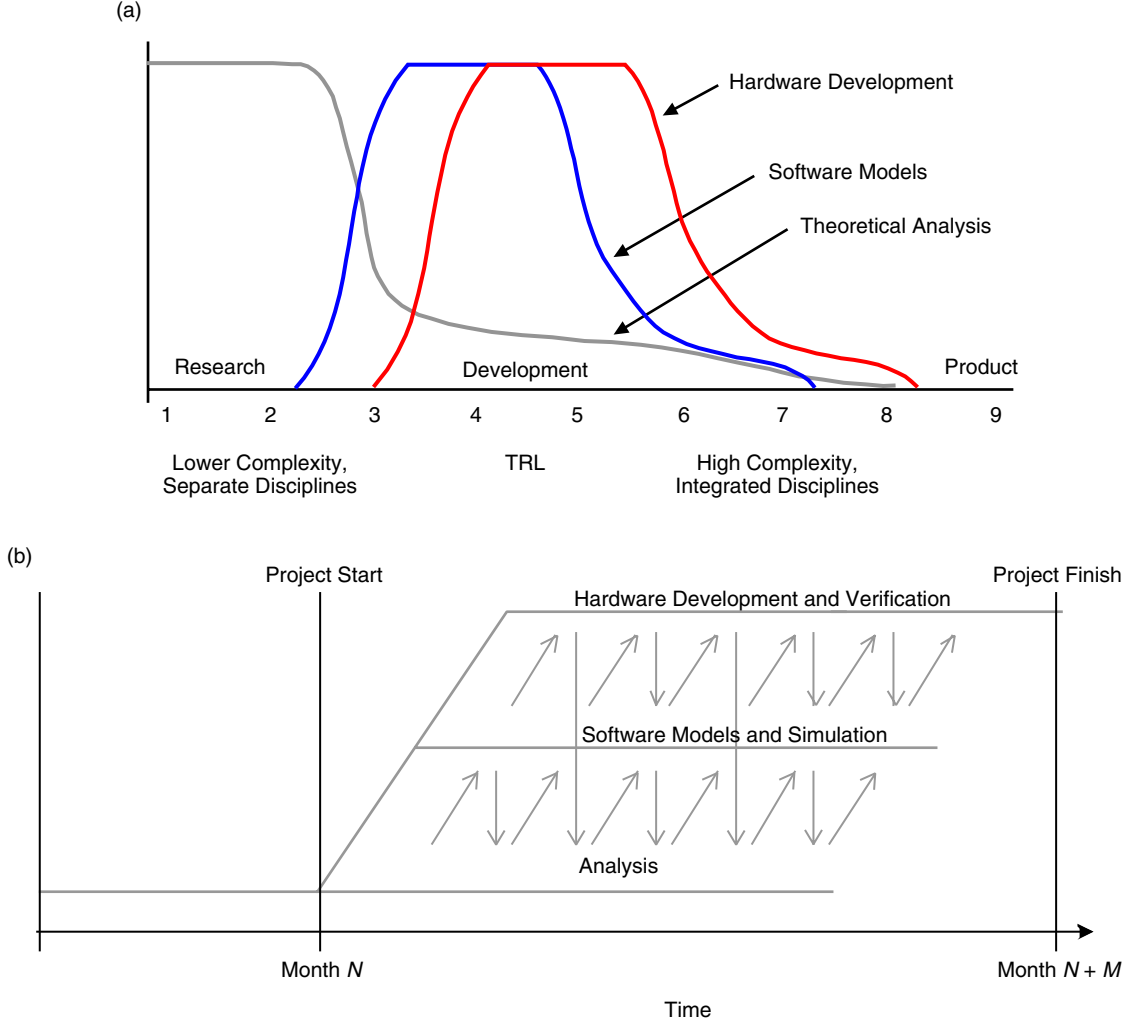
one or even a small number of factors. Answering design questions about VLSI systems as complex as those illustrated in Figs. 33, 37, and 40 usually is not about optimality of a criterion but about trading off many criteria against one another. One of the motives of Section VIII was to illustrate some of these trades. If the core problem posed in Section VIII (asynchronous parallel processing) had been done so and analyzed in purely mathematical terms, the most critical design questions would have been somewhat obscured. The asynchronous filter bank design is a complex undertaking with numerous trade-offs made between hardware complexity, rate reduction, the impact of filter design on communications systems performance (synchronization and decision processing), and commercially available methods for processing asynchronous digital data (many of which are device dependent).

The focus in this work was not on the analysis of performance of optical communications systems, although the architectures presented are capable of closely approximating theoretical performance of certain continuous-time systems. This assertion is given without mathematical proof, but basic simulation results of a software model were presented. In addition, there are numerous examples of communications algorithms and theoretical constructs based on similar signal processing theories and structures [8,9]. The performance analysis of the parallel architectures presented here is ongoing and will incorporate the numerous characteristics of the deep-space optical communications channel, including the primary transmitter and front-end detector characteristics. Before outlining the further design and performance analysis tasks to be conducted as part of the development, it is appropriate to consider the context in which this work will be performed. To communicate this context, the motivation for the development approach advocated by these authors also must be summarized.

First, it must be understood that there is tremendous flexibility in the development approach proposed. This approach is largely based on modern commercially defined and developed VLSI software tools, techniques, and best practices, and on the numerous very powerful FPGA devices currently available commercially. Development of the parameterized architectures presented in this work was one step in a development process composed of many complex and interacting processes and involving personnel from various disciplines of engineering. Figures 43(a) and 43(b) illustrate the overall development process possible with modern devices. Note in particular the overlapping tasks of analysis and hardware and software design. If this is not done, the hypothetical development time illustrated in Fig. 43(b) can easily double. The parallel analysis/software/hardware development is more feasible now than ever before due to reconfigurable hardware and associated modern software tools and processes.

Figure 44 illustrates the development path in greater detail; the state-of-the-art hardware development process is given with emphasis on VLSI circuit realization in FPGA devices. These devices have revolutionized modern hardware design. In addition to the obvious advantages of reconfigurable hardware, the stringent risk reduction and management techniques required by historical methods of VLSI implementation (analog and digital), particularly application-specific integrated circuits, are significantly relaxed. This results in a significantly altered design process. In addition, the inclusion of a development feedback path is possible, as depicted in the figure; such paths were often not feasible in VLSI design and implementation before the advent of the FPGA.

This process, if used correctly and in conjunction with a well designed hardware platform, facilitates rapid generation of prototypes targeting specific applications. If the application changes, the design may be iterated. *Arguably the largest advantage of this process targeting reconfigurable devices is the ability to make forward progress without a full set of requirements or without establishing values for all operational parameters. This is often the case with new or untested complex systems, where a combination of analysis and experimental physical work must be accomplished in conjunction in order to advance the state of the art.* However, a certain set of design parameters must be established to properly design the hardware platform; if not commercially available, this platform design itself requires another design process with many steps whose description is beyond the scope of this work. Certainly such things as A/D selection, FPGA selection, maximum clock rates and clock rate ranges, and a host of other parameters must be determined before the platform design and implementation or commercial purchase is made in the event that a commercial solution is available.

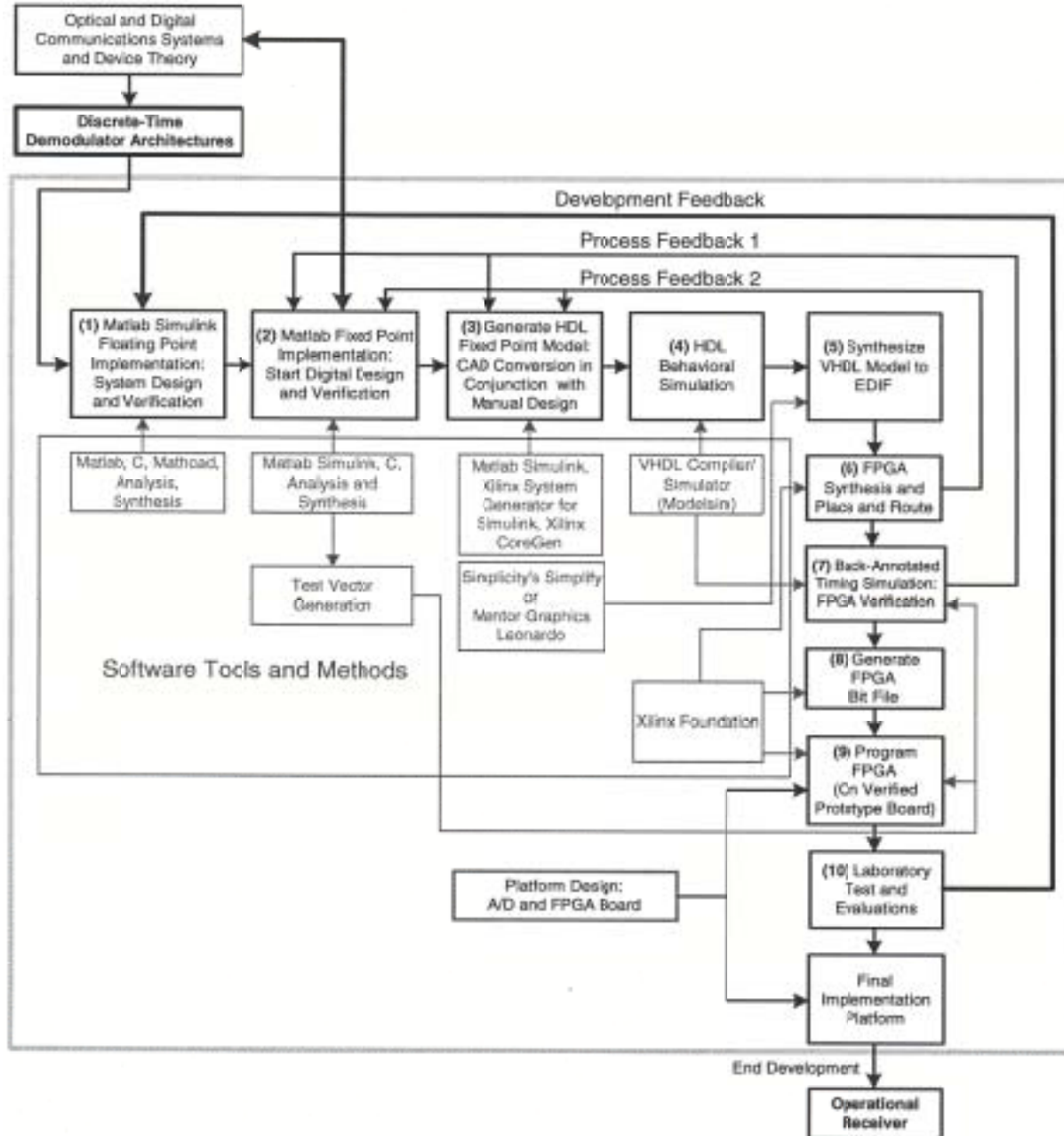


**Fig. 43. Development models: (a) demodulator development subsystems TRL model and (b) parallel development model.**

The development in this work was presented at a high enough level to address the most significant challenges of the all-digital design and implementation. There were many omissions in the work presented here that must be addressed as the development of the demodulator and complete receiver progresses. The primary issues of the demodulator development in the pre-hardware laboratory test phase, many of which should be addressed targeting a specific set of requirements, are summarized below.

- (1) The architectures presented may be further developed to incorporate numerous simplifications; many of these simplifications occur in a complex trade space of performance and implementation device limitations. Of particular interest is minimizing the complexity of the filter bank  $F_{N,p}^D(z, u, i)$ .
- (2) The function of the automatic gain control (AGC) was largely ignored in this work. This circuit may be accomplished in analog or digital circuits or both. In many systems, a “coarse” AGC circuit is used to control the amplitude of the signal input to the A/D to avoid clipping and underflow, while a “fine” AGC is implemented digitally to provide tight control of amplitude for such signal processing systems as the slot-synchronization loop, whose stability, bandwidth, damping factor, and overall performance depend on amplitude statistics.





**Fig. 44. State-of-the-art design process: the path from concept to operational digital demodulator.**

- (3) The completed discrete-time demodulator architecture must be converted to include quantized amplitude. This typically involves varying degrees of analysis and simulation.
- (4) The performance and type, open or closed loop, of the discrete-time slot-synchronization algorithm must be established for various receiver regions of operation using analysis techniques outlined in references presented in this work.
- (5) When a closed-loop system is derived from variations of Fig. 8 and the references given in Section II.B, the required analysis may be broken down into a number of subtasks:

- (a) The discrete-time loop design equation must be established, and gain margin, bandwidth, damping factor, and many other characteristics all must be determined.
  - (b) The synchronization performance with signal amplitude fluctuation characteristic of the optical channel must be characterized.
  - (c) Tracking versus acquisition performance must be established for the operational scenario or scenarios. This will include development of an acquisition algorithm. Among other things, this algorithm might determine when to switch loop filter bandwidths autonomously.
  - (d) Slot lock-detection circuits and performance must be established. The probability of false lock and false alarm must be established for these circuits.
  - (e) Performance in the presence of transmitter and receiver oscillator jitter must be established.
  - (f) A simple interpolation filter may be used with the system of Figs. 35 through 40. However, the slot or interpolation filter might be designed to achieve specified performance of the PPM slot-synchronization loop. The design might be further refined to incorporate criteria derived from the optimal detection given the optical channel.
- (6) Symbol-synchronization performance and lock detection must be established. A blind or pilot sequence system or both may be implemented.
- (7) Complete end-to-end system Monte-Carlo or pseudo-Monte-Carlo performance modeling and simulation of the demodulator must be performed. This type of comprehensive validation is accomplished in phases, starting with basic subsystem validation and progressively including more and more subsystems and channel parameters. This type of testing and validation accomplishes two goals: first system performance may be determined or approximated when doing so analytically is not possible or not feasible, and second cost and schedule risks of validating an operational system in the field are reduced. To first order, the resources that are spent on these end-to-end simulations are determined by the level of performance prediction required, the number of operational scenarios, and the risk tolerance of the project. As an example, before a system as complex as that of Fig. 37 is field tested, thousands of hours of simulations will be run to accomplish these goals even with moderate risk reduction goals over a moderate number of operational scenarios (a few dozen); many times that number may be utilized for more thorough validation across a more complex set of operating scenarios. These test and verification plans are integrated into the process of Fig. 42.
- (8) More optimal slot-synchronization algorithms might be developed. In addition to the decision-directed slot and symbol loops, different slot-synchronization algorithms might be derived based on an open-loop optimality criterion. It is envisioned that such algorithms may be implemented in the architecture given in Fig. 33, 37, or 40 with design evolution (it is envisioned that the majority of the architecture will remain unchanged).
- (9) Modern forward error-correction (FEC) decoders may require preprocessing; the performance of this preprocessing must be established.

This is a grossly simplified list of tasks that must be completed as part of the development of the parallel digital demodulator for free-space optical PPM. Before comprehensive laboratory testing and validation of the complete demodulator is undertaken, the majority of these tasks should be completed. However, the process of Fig. 42 is flexible and gives project managers many options for trading development schedule and progress with risk. It is possible and in many cases highly desirable to largely complete certain stand-alone signal processing subsystems, that is, to step them through the entire process of Fig. 42, and

to verify that subsystem with a laboratory hardware test. Such “front-runner” efforts serve a valuable process-risk-reduction role by validating the applicability and suitability of various vendor tools and software tool upgrades as well as hardware devices and platforms. Such front-runner activities also serve to demonstrate and validate the end-to-end concept-to-hardware process.

## References

- [1] R. Gagliardi and S. Karp, *Optical Communications*, Second Edition, New York: John Wiley and Sons, Inc., 1995.
- [2] V. Vilnrotter, A. Biswas, W. Farr, D. Fort, and E. Sigman, “Design and Analysis of a First-Generation Optical Pulse-Position Modulation Receiver,” *The Interplanetary Network Progress Report 42-148, October–December 2001*, Jet Propulsion Laboratory, Pasadena, California, pp. 1–20, February 15, 2002.  
[http://ipnpr.jpl.nasa.gov/tmo/progress\\_report/42-148/148K.pdf](http://ipnpr.jpl.nasa.gov/tmo/progress_report/42-148/148K.pdf)
- [3] T.-Y. Yan and C.-C. Chan, “Design and Development of Deep Space Baseline Optical Transceiver,” *Proc. SPIE*, vol. 3615, San Jose, California, January 1999.
- [4] J. Proakis, *Digital Communications*, New York: McGraw-Hill, Inc., 1995.
- [5] M. Simon, S. Hinedi, and W. Lindsey, *Digital Communication Techniques*, Englewood Cliffs, New Jersey: PTR Prentice Hall, 1995.
- [6] N. H. E. Weste and K. Eshraghian, *Principles of CMOS VLSI Design*, Reading, Massachusetts: Addison-Wesley Publishing Company, 1993.
- [7] W. J. Roesch, “GaAs Company Q&R Benchmarking Results,” *Proceedings GaAs Reliability Workshop*, October 1997.
- [8] R. Sadr, P. P. Vaidyanathan, D. Raphaeli, and S. Hinedi, *Parallel Digital Modem Using Multirate Filter Banks*, JPL Publication 94-20, Jet Propulsion Laboratory, Pasadena, California, August 1994.
- [9] A. Gray, *Very Large Scale Integration Architecture for Nyquist Rate Digital Communications Receivers*, Ph.D. Dissertation, University of Southern California, Los Angeles, May 2000.
- [10] R. McDonough and A. Whalen, *Detection of Signals in Noise*, New York: Academic Press, 1995.
- [11] C. W. Helstrom, *Elements of Signal Detection and Estimation*, Englewood Cliffs, New Jersey: PTR Prentice Hall, 1995.
- [12] H. L. Van Trees, *Detection, Estimation, and Modulation Theory, Part I*, New York: John Wiley and Sons, Inc., 1968.
- [13] R. Sadr and W. J. Hurd, “Detection of Signals by the Digital Integrate-and-Dump Filter With Offset Sampling,” *Telecommunications and Data Acquisition Progress Report 42-91, July–September 1987*, Jet Propulsion Laboratory, Pasadena, California, pp. 158–173, November 15, 1987.  
[http://tmo.jpl.nasa.gov/tmo/progress\\_report/42-91/91P.PDF](http://tmo.jpl.nasa.gov/tmo/progress_report/42-91/91P.PDF)
- [14] A. V. Oppenheim and R. W. Schaffer, *Discrete-Time Signal Processing*, Englewood Cliffs, New Jersey: Prentice-Hall, 1989.

- [15] P. P. Vaidyanathan, *Multirate Systems and Filter Banks*, Englewood Cliffs, New Jersey: Prentice-Hall, 1993.
- [16] H. Meyr and G. Ascheid, *Synchronization in Digital Communications*, Wiley Series in Telecommunications, New York: John Wiley and Sons, 1990.
- [17] C. N. Georgiades, "Detecting Random PPM Sequences in the Absence of Sequences in the Absence of Synchronization," *Proceedings of the 22nd Conference on Information Science Systems*, Princeton, New Jersey, pp. 159–163, March 1988.
- [18] X. Sun and F. Davidson, "Timing Recovery in Free Space Direct Detection Optical Communication Systems with PPM Signaling," *Communications*, 1989, ICC 89, BOSTON/ICC/89, *IEEE International Conference on World Prosperity Through Communications*, vol. 1, pp. 428–432, June 1989.
- [19] F. Davidson and X. Sun, "Slot Clock Recovery in Optical PPM Communication Systems with Avalanche Photodiode Photodetectors," *IEEE Transactions on Communications*, vol. 37, issue 11, pp. 1164–1172, November 1989.
- [20] X. Sun and F. Davidson, "Word Timing Recovery in Direct Detection Optical PPM Communication Systems with Avalanche Photodiodes Using a Phase Lock Loop," *IEEE Transactions on Communications*, vol. 38, issue 5, pp. 666–673 May 1990.
- [21] S. Chauchin, L.-Y. Huang, J.-J. Lee, and C.-K. Wang, "A Frame-Based Symbol Timing Recovery for Large Pull-In Range and Small Steady State Variation," *The First IEEE Asia Pacific Conference on ASICs*, AP-ASIC '99, pp. 75–78, August 1999.
- [22] G. L. Lui, "PPM Symbol Synchronization in Random Data," *IEEE Military Communications Conference, MILCOM '91, Military Communications in a Changing World*, vol. 3, pp. 1047–1053, November 1991.
- [23] C. N. Georgiades, "On the Synchronizability and Detectability of Random PPM Sequences," *IEEE Transactions on Information Theory*, vol. 35, issue 1, pp. 146–156, January 1989.
- [24] C. N. Georgiades, "On PPM Sequences with Good Autocorrelation Properties," *IEEE Transactions on Information Theory*, vol. 34, issue 3, pp. 571–576, May 1988.
- [25] R. Velidi and C. N. Georgiades, "Symbol Synchronization for Optical Multi-Pulse Pulse Position Modulation Systems," *IEEE International Conference on Personal Wireless Communications*, pp. 182–184, August 1994.
- [26] R. Velidi and C. N. Georgiades, "Optimal and Suboptimal Frame Synchronizers for Optical Multi-Pulse PPM," *IEEE International Symposium on Information Theory*, p. 79, July 1994.
- [27] K. Sato, T. Ohtsuki, I. Sasase, and S. Mori, "Performance Analysis of  $(m, 2)$  MPPM with Imperfect Slot Synchronization," *IEEE Pacific Rim Conference on Communications, Computers and Signal Processing*, vol. 2, pp. 765–768, May 1993.
- [28] M. Asano, T. Ohtsuki, H. Uehara, and I. Sasase, "A Novel Frame Synchronization Rule for Optical DPPM Systems with Restriction of Frame Length," *IEEE Global Telecommunications Conference, Communications: The Key to Global Prosperity*, vol. 2, pp. 923–927, November 1996.

- [29] D. Jinsong, I. Oka, and C. Fujiwara, "Symbol Error Probability of Time Spread PPM Signals in the Presence of Interference," *IEEE Pacific Rim Conference on Communications, Computers and Signal Processing, '10 Years PACRIM 1987-1997—Networking the Pacific Rim*, vol. 1, pp. 1–4, August 1997.
- [30] C. Georgiades, "Optimum Joint Slot and Symbol Synchronization for the Optical PPM Channel," *IEEE Transactions on Communications*, vol. 35, issue 6, pp. 632–636, June 1987.
- [31] K. Sato, T. Ohtsuki, H. Uehara, and I. Sasase, "Communication Systems with PPM Signaling Lightwave Technology," *Journal of Lightwave Technology*, vol. 14, issue 9, pp. 1963–1969, September 1996.
- [32] J. M. H. Elmirghani and R. A. Cryan, "Jitter Implications on Optical Fibre PPM Performance," *ICC 94, SUPERCOMM/ICC '94, Conference Record, IEEE International Conference on Serving Humanity Through Communications*, vol. 2, pp. 665–669, May 1994.
- [33] K. Sato, T. Ohtsuki, H. Uehara, and I. Sasase, "Communication Systems with PPM Signaling Lightwave Technology," *Journal of Lightwave Technology*, vol. 14, issue 9, pp. 1963–1969, September 1996.
- [34] V. A. Vilnrotter, E. R. Rodemich, and H. H. Tan, "A Synchronization Technique for Optical PPM Signals," *The Telecommunications and Data Acquisition Progress Report 42-87, July–September 1986*, Jet Propulsion Laboratory, Pasadena, California, pp. 24–31, November 15, 1986.  
[http://tmo.jpl.nasa.gov/tmo/progress\\_report/42-87/87C.PDF](http://tmo.jpl.nasa.gov/tmo/progress_report/42-87/87C.PDF)
- [35] S. Aguirre, W. J. Hurd, R. Kumar, and J. Statman, "A Comparison of Methods for DPLL Loop Filter Design," *The Telecommunications and Data Acquisition Progress Report 42-87, July–September 1986*, Jet Propulsion Laboratory, Pasadena, California, pp. 114–124, November 15, 1986.  
[http://tmo.jpl.nasa.gov/tmo/progress\\_report/42-87/87M.PDF](http://tmo.jpl.nasa.gov/tmo/progress_report/42-87/87M.PDF)
- [36] S. A. Stephens and J. B. Thomas, "Controlled-Root Formulation for Digital Phase-Locked Loops," *IEEE Transactions on Aerospace and Electronic Systems*, vol. 31, no. 1, January 1995.
- [37] A. Gray, "Parallel Sub-Convolution Filter Bank Architectures," *IEEE International Symposium on Circuits and Systems*, Bangkok, Thailand, May 2003.
- [38] J. Vollmer, "Analysis and Design of Numerically Controlled Oscillators Based on Linear Time-Variant Systems," *Proceedings of the IEEE-SP International Symposium on Time-Frequency and Time-Scale Analysis*, pp. 453–456, October 1998.
- [39] M. M. Al-Ibrahim, "A Multifrequency Range Digital Sinusoidal Oscillator with High Resolution and Uniform Frequency Spacing," *IEEE Transactions on Circuits and Systems II: Analog and Digital Signal Processing*, vol. 48, issue 9, pp. 872–876, September 2001.
- [40] M. M. Al-Ibrahim, "A Simple Recursive Digital Sinusoidal Oscillator with Uniform Frequency Spacing," *IEEE International Symposium on Circuits and Systems*, vol. 2, pp. 689–692, May 2001.